M 3

```
PPPPPPPPPPPP     RRRRRRRRRRRR     TTTTTTTTTTTTTTT     SSSSSSSSSSSS  MMM         MMM  BBBBBBBBBBBB
PPPPPPPPPPPP     RRRRRRRRRRRR     TTTTTTTTTTTTTTT     SSSSSSSSSSSS  MMM         MMM  BBBBBBBBBBBB
PPPPPPPPPPPP     RRRRRRRRRRRR     TTTTTTTTTTTTTTT     SSSSSSSSSSSS  MMM         MMM  BBBBBBBBBBBB
PPP         PPP  RRR         RRR        TTT        SSS              MMMMMM   MMMMMM  BBB         BBB
PPP         PPP  RRR         RRR        TTT        SSS              MMMMMM   MMMMMM  BBB         BBB
PPP         PPP  RRR         RRR        TTT        SSS              MMMMMM   MMMMMM  BBB         BBB
PPP         PPP  RRR         RRR        TTT        SSS              MMM   MMM   MMM  BBB         BBB
PPP         PPP  RRR         RRR        TTT        SSS              MMM   MMM   MMM  BBB         BBB
PPP         PPP  RRR         RRR        TTT        SSS              MMM   MMM   MMM  BBB         BBB
PPPPPPPPPPPP     RRRRRRRRRRRR           TTT           SSSSSSSSS     MMM         MMM  BBBBBBBBBBBB
PPPPPPPPPPPP     RRRRRRRRRRRR           TTT           SSSSSSSSS     MMM         MMM  BBBBBBBBBBBB
PPPPPPPPPPPP     RRRRRRRRRRRR           TTT           SSSSSSSSS     MMM         MMM  BBBBBBBBBBBB
PPP              RRR  RRR               TTT                    SSS  MMM         MMM  BBB         BBB
PPP              RRR  RRR               TTT                    SSS  MMM         MMM  BBB         BBB
PPP              RRR  RRR               TTT                    SSS  MMM         MMM  BBB         BBB
PPP              RRR      RRR           TTT                    SSS  MMM         MMM  BBB         BBB
PPP              RRR      RRR           TTT                    SSS  MMM         MMM  BBB         BBB
PPP              RRR      RRR           TTT                    SSS  MMM         MMM  BBB         BBB
PPP              RRR         RRR        TTT        SSSSSSSSSSSS     MMM         MMM  BBBBBBBBBBBB
PPP              RRR         RRR        TTT        SSSSSSSSSSSS     MMM         MMM  BBBBBBBBBBBB
PPP              RRR         RRR        TTT        SSSSSSSSSSSS     MMM         MMM  BBBBBBBBBBBB
```

M 7

```
**FILE**ID**DISPATCH

DDDDDDDD      IIIIII      SSSSSSSS  PPPPPPPP      AAAAAA    TTTTTTTTTT    CCCCCCCC  HH      HH
DDDDDDDD      IIIIII      SSSSSSSS  PPPPPPPP      AAAAAA    TTTTTTTTTT    CCCCCCCC  HH      HH
DD      DD      II      SS          PP      PP  AA      AA      TT      CC          HH      HH
DD      DD      II      SS          PP      PP  AA      AA      TT      CC          HH      HH
DD      DD      II      SS          PP      PP  AA      AA      TT      CC          HH      HH
DD      DD      II      SS          PP      PP  AA      AA      TT      CC          HH      HH
DD      DD      II        SSSSSS    PPPPPPPP    AA      AA      TT      CC          HHHHHHHHHH
DD      DD      II        SSSSSS    PPPPPPPP    AA      AA      TT      CC          HHHHHHHHHH
DD      DD      II              SS  PP          AAAAAAAAAA      TT      CC          HH      HH
DD      DD      II              SS  PP          AAAAAAAAAA      TT      CC          HH      HH
DD      DD      II              SS  PP          AA      AA      TT      CC          HH      HH    ....
DD      DD      II              SS  PP          AA      AA      TT      CC          HH      HH    ....
DDDDDDDD      IIIIII    SSSSSSSS    PP          AA      AA      TT        CCCCCCCC  HH      HH    ....
DDDDDDDD      IIIIII    SSSSSSSS    PP          AA      AA      TT        CCCCCCCC  HH      HH    ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

DI
VO

```
0 MODULE DISPATCH ( %TITLE 'Print Symbiont - main dispatch routines'
0         IDENT = 'V04-000',
0         ADDRESSING_MODE (EXTERNAL = GENERAL)
0         ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE  IS  HEREBY  *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY:
1 !       Print Symbiont.
1 !
1 ! ABSTRACT:
1 !       This module contains the main control loop for the symbiont.
1 !       PSM$DISPATCH steps through the various symbiont states and
1 !       switches among the input routines.  It also calls the format
1 !       and output service routines.
1 !
1 !       This module also contains various miscellaneous subroutines
1 !       related to error handling, checkpointing, and push/pop of input
1 !       routines.
1 !
1 ! ENVIRONMENT:
1 !       VAX/VMS user mode, AST-level.
1 !--
1
1 ! AUTHOR: G. Robert, CREATION DATE: 31-Aug-1982
1 !
1 ! MODIFIED BY:
1 !
1 !       3B-011  RRB3011         Rowland R. Bradley      09-Aug-1984
1 !               If aligning the file and READ_COMPLETION detects EOF
1 !               then send a response to job controller.  Added the
1 !               test for psm$v_align in READ_COMPLETION case of
```

```
1 !                     PSM$FUNCTION_DISPATCH.
1 !
1 !     3B-010  RRB3010         Rowland R. Bradley      27-Jul-1984
1 !             Clear the suppress_output bit and the search_for_page
1 !             bit on EOF (only on file service).  Also conditionally
1 !             set stop_page to -1 (only when the current service is
1 !             not nested).  This fixes the symbiont hang when search
1 !             for page is past end of file and the /HEADER & /PAGES
1 !             ill interaction.
1 !
1 !     3B-009  GRR3009         Gregory R. Robert       25-Jul-1984
1 !             Remove the global clear of the sequence bit in print
1 !             control.  This fixes the problem /header interfering
1 !             with line numbers.
1 !
1 !     3B-008  GRR3008         Gregory R. Robert       11-Jul-1984
1 !             Suppress leading carriage control for first record
1 !             of implied carriage control input service.  Remove
1 !             code that resets accounting totals after separation pages.
1 !
1 !     3B-007  GRR3007         Gregory R. Robert       16-May-1984
1 !             Defend against attempted CLOSE when service routine
1 !             is non-existent
1 !
1 !     3B-006  GRR3006         Gregory R. Robert       09-May-1984
1 !             Fix call interface for user filter/format routines.
1 !
1 !     3B-005  GRR3005         Gregory R. Robert       29-Apr-1983
1 !             FT2 bugfixes plus margins.
1 !
1 !     3B-004  GRR3004         Gregory R. Robert       01-Sep-1983
1 !             Enabled PHY_IO so that DCS escape sequences can be
1 !             written PASSALL or NOFORMAT.
1 !
1 !     3B-003  GRR3003         Gregory R. Robert       23-Aug-1983
1 !             Bugfixes, page_setup_modules, form_setup_modules,
1 !             sheet_feed, symbiont initiated pause_task and stop_stream,
1 !             hangup code, read and write item services
1 !
1 !     3B-002  GRR3002         Gregory R. Robert       03-Aug-1983
1 !             Rewrite for new design.
1 !
1 !     3B-001  GRR3001         Gregory R. Robert       29-Jul-1983
1 !             Created new module.
1 !
1 !
1 !**
```

```
1 LIBRARY 'SYS$LIBRARY:LIB';
1 REQUIRE 'LIB$:SMBDEF';
1 REQUIRE 'SRC$:SMBREQ';
1
1 EXTERNAL ROUTINE
1         PSM$ALLOCATE_DSB : NOVALUE,
1         PSM$ALLOCATE_IOB : NOVALUE,
1         PSM$DEALLOCATE_DSB : NOVALUE,
1         SMB$INITIALIZE,
1         PSM$RECEIVE_MESSAGE_AST,
1         PSM$SCHEDULE_NON_AST,
1         SMB$SEND_TO_JOBCTL,
1         PSM$WAIT_FOR_NON_AST
1         ;
1
1 EXTERNAL
1         PSM$GL_SCBVEC    : VECTOR,                    ! SCB index table
1         PSM$GL_MAXBUF    ,                            ! maximum output buffer size
1         PSM$GL_USER_CTX  ,                            ! user context area size
1         PSM$SRV : BLOCKVECTOR[,SRV_S_SRV, BYTE],      ! service routine table
1         PSM$XLATE_ALIGN : VECTOR [,BYTE],             ! MOVTUC table for X's and 9's
1         PSM$XLATE_8BIT  : VECTOR [,BYTE]              ! MOVTUC table for normal print
1         ;
1
1 LITERAL
1         EDIT_MASK = %B '110000'                       ! upcase and compact spaces and tabs
1         ;
```

```
1 FORWARD ROUTINE
1           PSM$FUNCTION_DISPATCH     : NOVALUE,       ! main control loop
1           PSM$REPORT                ,                ! async. event completion
1           PSM$INCLUDE_MODULES       ,                ! queues modules for insertion
1           PSM$PRINT                 ,                ! initialization entry point
1           PSM$STORE_ERRORS          ,                ! store errors for latter
1
1           ABORT_TASK                : NOVALUE,       ! aborts current file
1           CARRIAGE_CONTROL          ,                ! computes carriage control
1           ENQUEUE_CHECKPOINT        : NOVALUE,       ! save a checkpoint
1           EXPAND_CONDITION_VECTOR   : NOVALUE,       ! expand errors to text
1           FIND_CHECKPOINT           ,                ! find a checkpoint
1           GET_BUFFER                ,                ! get a buffer
1           HANDLER                   ,                ! main signal handler
1           PUTMSG_ACTION             ,                ! $PUTMSG action routine
1           RESUME_SERVICE            : NOVALUE,       ! POP input routine
1           SAVE_CHECKPOINT           : NOVALUE,       ! contstruct a checkpoint
1           SCHEDULE_SERVICE          ,                ! schedule an input routine
1           SEARCH_FOR_STRING         ,                ! look for a search string
1           SUSPEND_SERVICE           : NOVALUE,       ! PUSH input routine
1           STRIP_COMMA_DELIMITED_ITEM                 ! parse comma separated lists
1           ;
```

```
1 %SBTTL 'FUNCTION_DISPATCH - Main symbiont control loop'
1 ! Functional Description:
1 !                 Steps through symbiont states, switching among
1 !                 input routines and calling format/output service
1 !                 routines as necessary.
1 !
1 ! Formal Parameters:
1 !                 Address of a SCB (stream control block)
1 !
1 ! Implicit Inputs:
1 !                 none
1 !
1 ! Implicit Outputs:
1 !                 none
1 !
1 ! Returned Value:
1 !                 none
1 !
1 ! Side Effects:
1 !                 Asynchronous IO events may be initiated
1 !--
1 GLOBAL ROUTINE PSM$FUNCTION_DISPATCH (
1         SCB             : REF $BBLOCK           ! stream control block address
1         ) : NOVALUE =
2 BEGIN
2
2 LITERAL
2         FIRST_STATE                     = 0,            ! Must be zero
2         START_TASK                      = FIRST_STATE,
2         FIND_WORK                       = 1,
2         OPEN                            = 2,
2         OPEN_COMPLETION                 = 3,
2         READ                            = 4,
2         READ_COMPLETION                 = 5,
2         INPUT_FILTER                    = 6,
2         INPUT_FILTER_COMPLETION         = 7,
2         FORMAT                          = 8,
2         FORMAT_COMPLETION               = 9,
2         OUTPUT_FILTER                   = 10,
2         OUTPUT_FILTER_COMPLETION        = 11,
2         WRITE                           = 12,
2         WRITE_COMPLETION                = 13,
2         CLOSE                           = 14,
2         CLOSE_COMPLETION                = 15,
2         STOP_TASK                       = 16,
2         IDLE                            = 17,
2         RESUME                          = 18,
2         LAST_STATE                      = RESUME
2         ;
2
2 LITERAL
2         CONTINUE        = 1;
2
2 LABEL
2         CASE_STATEMENT;
2
2 ! For each state specify the default next_state
```

```
!
OWN
    NEXT_STATE : VECTOR [LAST_STATE + 1, BYTE]
        PSECT (CODE) PRESET (
            [START_TASK]                  = FIND_WORK,
            [FIND_WORK]                   = OPEN,
            [OPEN]                        = OPEN_COMPLETION,
            [OPEN_COMPLETION]             = READ,
            [READ]                        = READ_COMPLETION,
            [READ_COMPLETION]             = INPUT_FILTER,
            [INPUT_FILTER]                = INPUT_FILTER_COMPLETION,
            [INPUT_FILTER_COMPLETION]     = FORMAT,
            [FORMAT]                      = FORMAT_COMPLETION,
            [FORMAT_COMPLETION]           = OUTPUT_FILTER,
            [OUTPUT_FILTER]               = OUTPUT_FILTER_COMPLETION,
            [OUTPUT_FILTER_COMPLETION]    = WRITE,
            [WRITE]                       = WRITE_COMPLETION,
            [WRITE_COMPLETION]            = RESUME,
            [CLOSE]                       = CLOSE_COMPLETION,
            [CLOSE_COMPLETION]            = FIND_WORK,
            [STOP_TASK]                   = IDLE,
            [IDLE]                        = IDLE,
            [RESUME]                      = RESUME
        );

! Specify expected errors that do not cause automatic task abort
! on a state specific basis
!
OWN
    EXPECTED_ERRORS : VECTOR [LAST_STATE + 1]
        PSECT (CODE) PRESET (
            [READ_COMPLETION]             = PLIT (PSM$_EOF, RMS$_EOF),
            [FORMAT_COMPLETION]           = PLIT (PSM$_BUFFEROVF, PSM$_NEWPAGE,
                                                  PSM$_ESCAPE, PSM$_SUSPEND)
        );
```

```
2
2  ! Advance through the symbiont states until an asynchronous service
2  ! returns pending, or all output buffers are in use, or a pause is
2  ! requested by the job controller
2  !
2  UNTIL .SCB[PSM$L_SERVICE_STATUS] EQL PSM$_PENDING
2  DO
2      CASE_STATEMENT:
2      BEGIN
3      LOCAL SERVICE : REF $BBLOCK;          ! Table entry for current input service
3      LOCAL SERVICE_STATUS;                 ! Status of most recent service
3      LOCAL CURRENT_STATE;                  ! Current symbiont state
3
3
3      ! Don't do anything unless we have or can get an output buffer
3      !
3      IF .SCB[PSM$A_IOB] EQL 0
3      THEN
3          IF NOT GET_BUFFER  (.SCB)
3          THEN
3              RETURN;
3
3
3      ! Locate the current input service, pickup the last
3      ! service status, and initialize the next service status to success
3      !
3      SERVICE = PSM$SRV[.SCB[PSM$B_SERVICE_INDEX],0,0,0,0];
3      SERVICE_STATUS = .SCB[PSM$L_SERVICE_STATUS];
3      SCB[PSM$L_SERVICE_STATUS] = SS$_NORMAL;
3
3
3      ! Get the current state and select the next state default
3      !
3      CURRENT_STATE = .SCB[PSM$B_STATE];
3      SCB[PSM$B_STATE] = .NEXT_STATE[.CURRENT_STATE];
3
3
3      ! Report any unexpected errors
3      !
3      IF NOT .SERVICE_STATUS
3      THEN
4          BEGIN
4          BIND ERROR_LIST = .EXPECTED_ERRORS[.CURRENT_STATE] : VECTOR;
4          LOCAL EXPECTED_ERROR : INITIAL (0);
4
4          ! If an expected error list is specified for the current
4          ! state then loop through the list to see if the service
4          ! error is expected.
4          !
4          IF ERROR_LIST NEQ 0
4          THEN
4              INCRU ERROR_INDEX TO .ERROR_LIST[-1] - 1
4              DO
4                  IF .SERVICE_STATUS EQL .ERROR_LIST[.ERROR_INDEX]
4                  THEN
5                      BEGIN
5                      EXPECTED_ERROR = 1;
```

```
;  307          1250  5                       EXITLOOP;
;  308          1251  4                       END;
;  309          1252  4
;  310          1253  4             ! If an unexpected error then report it
;  311          1254  4             !
;  312          1255  4             IF NOT .EXPECTED_ERROR
;  313          1256  4             THEN
;  314          1257  4                 PSM$STORE_ERRORS (.SCB, .SERVICE_STATUS);
;  315          1258  3             END;
;  316          1259  3
;  317          1260  3
;  318          1261  3         ! Dispatch to the appropriate code
;  319          1262  3         !
;  320          1263  3         CASE .CURRENT_STATE FROM FIRST_STATE TO LAST_STATE OF
;  321          1264  3             SET
;  322          1265  3
;  323          1266  3
;  324          1267  3 ! NOTE: the usual VMS/Bliss formating conventions are altered here.
;  325          1268  3 !       Each case label begins a new page and is left justifed.
;  326          1269  3 !
```

```
3 [RESUME]:
4 BEGIN
4
4 ! ++
4 !
4 ! RESUME handles positioning, searching, and alignment requests.
4 ! The desired starting page is reached by successive approximations
4 ! utilizing the POSITION TO KEY and REWIND service functions and the
4 ! SEARCH_FOR_PAGE, SEARCH_FOR_STRING and ALIGN features of the
4 ! symbiont
4 !
4 ! --
4
4 LOCAL CHECKPOINT : REF $BBLOCK;
4
4 ! Reset positioning and alignment controls
4 !
4 SCB[PSM$A_XLATE_TABLE] = PSM$XLATE_8BIT;
4 SCB[PSM$V_ALIGN] = 0;
4 SCB[PSM$V_SEARCH_FOR_PAGE] = 0;
4 SCB[PSM$V_SEARCH_FOR_STRING] = 0;
4 SCB[PSM$V_SUPPRESS_OUTPUT] = 0;
4
4
4 ! If no start page specified then default to current page
4 !
4 IF .SCB[PSM$L_START_PAGE] EQL 0 THEN SCB[PSM$L_START_PAGE] = .SCB[PSM$L_PAGE];
4
4
4 ! Look for a useable checkpoint that improves on the current page location
4 !
4 CHECKPOINT = FIND_CHECKPOINT (.SCB);
4 IF .CHECKPOINT NEQ 0
4 THEN
5     BEGIN
5     LOCAL KEY_DESC : VECTOR [2];
5
5       ! Save the checkpoint address for INPUT_FILTER_COMPLETION
5       !
5       SCB[PSM$A_CHECKPOINT] = .CHECKPOINT;
5
5
5       ! Mark the next read as offset, set the new page number
5       ! and cancel any outstanding input record
5       !
5       SCB[PSM$V_READ_OFFSET] = 1;
5       SCB[PSM$L_PAGE] = .CHECKPOINT[SMBMSG$L_PAGE];
5       SCB_SIZE_ (INPUT_RECORD) = 0;
5
5
5       ! Set up the user key descriptor
5       !
5       KEY_DESC[0] = 4;
5       KEY_DESC[1] = CHECKPOINT[SMBMSG$Q_USER_KEY];
5
5
5       ! Request random positioning
```

```
5           !
5           SCB[PSM$L_SERVICE_STATUS] = BLISS (
5               .SERVICE[SRV_A_SERVICE],            !  - current input service
5               SCB,                                !  - SCB address by reference
5               SCB[PSM$R_USER_CONTEXT_AREA],       !  - user context area
5               UPLIT (PSM$K_POSITION_TO_KEY),      !  - POSITION_TO_KEY function
5               KEY_DESC,                           !  - checkpoint descriptor
5               0);                                 !  - <not used>
5
5
5           IF .SCB[PSM$L_SERVICE_STATUS] EQL PSM$_FUNNOTSUP
5           THEN
5               CODEERR_ ;        ! POSITION_TO_KEY is symetrical with GET_KEY
5
5
5           LEAVE CASE_STATEMENT;
4           END;
4
4
4       ! If the start page is still less than the current page then rewind
4       !
4       IF .SCB[PSM$L_START_PAGE] LSSU .SCB[PSM$L_PAGE]
4       THEN
5           BEGIN
5
5           ! Adjust the page context and cancel any outstanding input record
5           !
5           SCB[PSM$L_PAGE] = 1;
5           SCB[PSM$L_RECORD_NUMBER] = 0;
5           SCB_SIZE_ (INPUT_RECORD) = 0;
5           SCB[PSM$L_CARCON] = 0;
5
5
5           ! Request the input service to rewind
5           !
5           SCB[PSM$L_SERVICE_STATUS] = BLISS (
5               .SERVICE[SRV_A_SERVICE],            !  - current input service
5               SCB,                                !  - SCB address by reference
5               SCB[PSM$R_USER_CONTEXT_AREA],       !  - user context area
5               UPLIT (PSM$K_REWIND),               !  - REWIND function
5               0,                                  !  - <not used>
5               0);                                 !  - <not used>
5
5
5           IF .SCB[PSM$L_SERVICE_STATUS] EQL PSM$_FUNNOTSUP
5           THEN
5               CODEERR_ ;                          ! REWIND is a required function
5
5           LEAVE CASE_STATEMENT;
4           END;
4
4
4       ! If the start page is still forward of the current page then start page search
4       !
4       IF .SCB[PSM$L_START_PAGE] GTRU .SCB[PSM$L_PAGE]
4       THEN
5           BEGIN
```

```
5           SCB[PSM$L_STOP_PAGE] = .SCB[PSM$L_START_PAGE];
5           SCB[PSM$V_SEARCH_FOR_PAGE] = 1;
5           SCB[PSM$V_SUPPRESS_OUTPUT] = 1;
5           SCB[PSM$B_STATE] = FORMAT;
5           LEAVE CASE_STATEMENT;
4           END;
4
4
4   ! Set the stop page for string search or in case we start printing
4   !
4   SCB[PSM$L_STOP_PAGE] = -1;
4   IF .ITEM_PRESENT_ (LAST_PAGE)
4   THEN
4       SCB[PSM$L_STOP_PAGE] = .SCB[PSM$L_LAST_PAGE] + 1;
4
4
4   ! Start page reached -- initiate a string search if requested
4   !
4   IF TESTBITSC (ITEM_PRESENT_ (SEARCH_STRING))
4   THEN
5       BEGIN
5       BAS$EDIT (SCB[PSM$Q_SEARCH_STRING],SCB[PSM$Q_SEARCH_STRING], EDIT_MASK);
5       CLEAR_STRING_ (SCB[PSM$Q_SEARCH_CONTEXT]);
5       SCB[PSM$V_SEARCH_FOR_STRING] = 1;
5       SCB[PSM$V_SUPPRESS_OUTPUT] = 1;
5       SCB[PSM$B_STATE] = FORMAT;
5       LEAVE CASE_STATEMENT;
4       END;
4
4
4   ! Positioning complete -- check for alignment
4   !
4   IF TESTBITSC (ITEM_PRESENT_ (ALIGNMENT_PAGES))
4   THEN
5       BEGIN
5       SCB[PSM$V_ALIGN] = 1;
5       IF .REQUEST_FLAG_ (ALIGNMENT_MASK)
5       THEN
5           SCB[PSM$A_XLATE_TABLE] = PSM$XLATE_ALIGN;
5       SCB[PSM$L_STOP_PAGE] = .SCB[PSM$L_PAGE] + .SCB[PSM$L_ALIGNMENT_PAGES];
5       SCB[PSM$B_STATE] = FORMAT;
5
5       ! (Since we don't alter SCB[PSM$L_START_PAGE] repositioning to
5       ! the current page following alignment completion is automatic).
5       !
5       LEAVE CASE_STATEMENT;
4       END;
4
4
4   ! Print only one page if in sheet_feed mode
4   !
4   IF .$BBLOCK [SCB[PSM$L_PRINT_CONTROL], SMBMSG$V_SHEET_FEED]
4   THEN
4       SCB[PSM$L_STOP_PAGE] = .SCB[PSM$L_PAGE] + 1;
4
4
4   ! Resume complete -- tell the job controller
```

```
4 !
4 SMB$SEND_TO_JOBCTL (
4       SCB[PSM$L_STREAM_INDEX],              ! - stream number
4       SCB[PSM$L_REQUEST_RESPONSE]);         ! - responding to resume or start task
4
4
4 ! If pause at completion was requested then marks as pending
4 !
4 IF TESTBITSC (REQUEST_FLAG_ (PAUSE_COMPLETE))
4 THEN
5       BEGIN
5       SCB[PSM$V_RESUME_WAIT] = 1;
5       SCB[PSM$L_SERVICE_STATUS] = PSM$_PENDING;
5       END
4 ELSE
4       SCB[PSM$B_STATE] = FORMAT;
4
3 END;
```

```
3 [START_TASK]:
4 BEGIN
4
4 ! Tell the job controller that START_TASK is complete and we
4 ! are now printing
4 !
4 SMB$SEND_TO_JOBCTL (
4     SCB[PSM$L_STREAM_INDEX],            ! - stream number
4     SCB[PSM$L_REQUEST_RESPONSE]);       ! - responding to start task
4
4
4 ! If pause at completion was requested then marks as pending
4 !
4 IF TESTBITSC (REQUEST_FLAG_ (PAUSE_COMPLETE))
4 THEN
5     BEGIN
5     SCB[PSM$V_RESUME_WAIT] = 1;
5     SCB[PSM$L_SERVICE_STATUS] = PSM$_PENDING;
4     END;
4
3 END;
```

```
3 [FIND_WORK]:
4 BEGIN
4
4 ! If we are stopping the stream (STOP/NEXT or STOP/RESET) then stop
4 ! the task
4 !
4 IF .SCB[PSM$V_RESET]
4 THEN
4     SCB[PSM$B_STATE] = STOP_TASK
4 ELSE
4     ! Otherwise look for an input service
4     !
4     IF NOT SCHEDULE_SERVICE (.SCB)
4     THEN
4         ! None found, cancel sheet_feed and flush the output stream
4         !
5         BEGIN
5         $BBLOCK [SCB[PSM$L_PRINT_CONTROL], SMBMSG$V_SHEET_FEED] = 0;
5         $BBLOCK [.SCB[PSM$A_IOB], IOB_V_FLUSH_PENDING] = 1;
5         SCB[PSM$B_STATE] = OUTPUT_FILTER;
5         END;
4
3 END;
```

```
3 [OPEN]:
4 BEGIN
4
4 ! If resuming a suspended service then continue at FORMAT
4 !
4 IF .BITVECTOR [SCB[PSM$L_SERVICE_OPEN], .SCB[PSM$B_SERVICE_INDEX]]
4 THEN
5     BEGIN
5     SCB[PSM$B_STATE] = FORMAT;
5     LEAVE CASE_STATEMENT;
4     END;
4
4 ! Establish the default carriage control
4 !
4 SCB[PSM$L_FUNCTION_ARGUMENT] = PSM$K_CC_IMPLIED;
4
4
4 ! Tell the input service to OPEN
4 !
4 SCB[PSM$L_SERVICE_STATUS] = BLISS (
4     .SERVICE[SRV_A_SERVICE],              !  - current input service
4     SCB,                                  !  - SCB address by reference
4     SCB[PSM$R_USER_CONTEXT_AREA],         !  - user context area
4     UPLIT (PSM$K_OPEN),                   !  - OPEN function
4     SCB[PSM$Q_FILE_SPECIFICATION],        !  - file name
4     SCB[PSM$L_FUNCTION_ARGUMENT]);        !  - receives carriage control type
4
3 END;
```

```
3 [OPEN_COMPLETION]:
4 BEGIN
4
4 ! If the open failed then look for more work
4 !
4 !
4 IF NOT .SERVICE_STATUS
4 THEN
5     BEGIN
5     SCB[PSM$B_STATE] = FIND_WORK;
5     LEAVE CASE_STATEMENT;
4     END;
4
4
4 ! Mark the service OPEN and set the carriage control type
4 !
4 BITVECTOR [SCB[PSM$L_SERVICE_OPEN], .SCB[PSM$B_SERVICE_INDEX]] = 1;
4 SCB[PSM$B_CC_TYPE] = .SCB[PSM$L_FUNCTION_ARGUMENT];
4
4 ! If this service is NOT a nested service then init the stop page
4 ! to default of -1(end of file).
4 !
4 IF .SCB[PSM$B_INPUT_DEPTH] LEQ 0
4 THEN
4     SCB[PSM$L_STOP_PAGE] = -1;
4
4 ! Handle special features of main file processing including
4 ! checkpoint restarts, first and last page (/PAGE=(first,last))
4 ! and print flags (/FEED, /HEADER, /SPACE)
4 !
4 IF .SERVICE[SRV_B_SERVICE_TYPE] EQL SRV_K_FILE_SERVICE
4 THEN
5     BEGIN
5
5     ! Set the print flags
5     !
5     SCB[PSM$L_PRINT_FLAGS] = .SCB[PSM$L_PRINT_CONTROL];
5
5     ! Set up the local top and left margins (PSM$MAIN_FORMAT
5     ! uses the global right and bottom margins because, with
5     ! /wrap, /truncate, /feed disabled they have no effect.
5     !
5     SCB[PSM$L_L_MARGIN] = .SCB[PSM$L_LEFT_MARGIN];
5     SCB[PSM$L_T_MARGIN] = .SCB[PSM$L_TOP_MARGIN];
5
5     ! Supress sequence numbers if width is too small
5     !
6     IF (.SCB[PSM$L_FORM_WIDTH] - .SCB[PSM$L_LEFT_MARGIN])
5         - .SCB[PSM$L_RIGHT_MARGIN] LSSU 8
5     THEN
5         PRINT_FLAG_ (SEQUENCED) = 0;
5
5
5     ! If restarting from a checkpoint, or if a first page was
5     ! specified, then setup so that the RESUME processing will
5     ! position to the correct page.
5     !
```

```
5       IF .ITEM_PRESENT_ (CHECKPOINT_DATA)
5       THEN
5           ! Checkpoint -- save it if valid
6           BEGIN
6           BIND CKP = .SCB_ADDR_ (CHECKPOINT_DATA) : $BBLOCK;
6           IF .CKP[SMBMSG$B_CHECKPOINT_LEVEL] EQL SMBMSG$K_STRUCTURE_LEVEL
6           THEN
7               BEGIN
7               ENQUEUE_CHECKPOINT (.SCB, SCB[PSM$Q_CHECKPOINT_DATA]);
7               SCB[PSM$L_START_PAGE] = .CKP[SMBMSG$L_PAGE];
7               END
6           END
5       ELSE
5           ! /PAGE=(first_page,"")
5           !
5           IF .ITEM_PRESENT_ (FIRST_PAGE)
5           THEN
5               SCB[PSM$L_START_PAGE] = .SCB[PSM$L_FIRST_PAGE];
5
5       ! Flush the output stream -- positioning will be picked up
5       ! after flush is complete
5       !
5       $BBLOCK [.SCB[PSM$A_IOB],IOB_V_FLUSH_PENDING] = 1;
5       SCB[PSM$B_STATE] = OUTPUT_FILTER;
4       END;
3   END;
```

```
3 [READ]:
4 BEGIN
4
4 ! Initialize the user record descriptor (dynamic)
4 !
4 CLEAR_STRING_ (SCB[PSM$Q_USER_RECORD]);
4
4
4 ! Quit if input service ended
4 !
4 IF .SCB[PSM$V_EOF] THEN LEAVE CASE_STATEMENT;
4
4
4 ! Clear the record header field and set the new_record flag
4 !
4 SCB[PSM$L_RECORD_HEADER] = 0;
4 SCB[PSM$V_NEW_RECORD] = 1;
4
4
4 ! Defend against an attempt to READ a non-existent service
4 !
4 IF .SERVICE[SRV_A_SERVICE] EQL 0
4 THEN
5     BEGIN
5     SERVICE_STATUS = PSM$_FUNNOTSUP;
5     LEAVE CASE_STATEMENT;
4     END;
4
4
4 ! Initiate the READ
4 !
4 SCB[PSM$L_SERVICE_STATUS] = BLISS (
4     .SERVICE[SRV_A_SERVICE],            !  - current input service
4     SCB,                                !  - SCB address by reference
4     SCB[PSM$R_USER_CONTEXT_AREA],       !  - user context area
4     UPLIT (PSM$K_READ),                 !  - READ function
4     SCB[PSM$Q_USER_RECORD],             !  - quadword to receive desc
4     SCB[PSM$L_RECORD_HEADER]);          !  - record header
4
3 END;
```

```
3 [READ_COMPLETION]:
4 BEGIN
4
4 ! Check for exceptions
4 !
4 IF NOT .SERVICE_STATUS
4 OR .SCB[PSM$V_EOF]
4 OR .SERVICE_STATUS EQL PSM$_FUNNOTSUP
4 THEN
5     BEGIN
5
5     ! Assume we will close
5     !
5     SCB[PSM$B_STATE] = CLOSE;
5
5     ! If EOF and searching for page then disable suppression and page
5     ! search.
5     !
6     IF  (.SERVICE_STATUS EQL PSM$_EOF OR .SERVICE_STATUS EQL RMS$_EOF)
6     AND (.SCB[PSM$V_SEARCH_FOR_STRING] OR .SCB[PSM$V_SEARCH_FOR_PAGE]
6           OR .SCB[PSM$V_ALIGN])
5     THEN
6         BEGIN
6         ! Only if this is a file service EOF do we wish to stop
6         ! searching
6         !
6         IF .SERVICE[SRV_B_SERVICE_TYPE] EQL SRV_K_FILE_SERVICE
6         THEN
7             BEGIN
7             SCB[PSM$V_SUPPRESS_OUTPUT] = 0;
7             SCB[PSM$V_SEARCH_FOR_PAGE] = 0;
6             END;
6
6         ! If EOF encountered while searching and resuming (NOT start_task)
6         ! then report it and pause the thread
6         !
6         IF .SCB[PSM$L_REQUEST_RESPONSE] EQL SMBMSG$K_RESUME_TASK
6         THEN
7             BEGIN
7             SMB$SEND_TO_JOBCTL (
7                 SCB[PSM$L_STREAM_INDEX],            ! - stream number
7                 SCB[PSM$L_REQUEST_RESPONSE],        ! - request response (resume)
7                 0,                                  ! - no accounting
7                 0,                                  ! - no checkpoint
7                 0,                                  ! - no device status
7                 SERVICE_STATUS                      ! - report the error (eof)
7                 );
7             SCB[PSM$B_STATE] = RESUME;
7             SCB[PSM$V_RESUME_WAIT] = 1;
7             SCB[PSM$L_SERVICE_STATUS] = PSM$_PENDING;
6             END;
6
6         LEAVE CASE_STATEMENT;
5         END;
4     END;
4
4
```

```
4
4
4 ! Update accounting and current record number
4 !
4 INCREMENT_ (ACC_DATA_ (RMS_GETS));
4 INCREMENT_ (SCB[PSM$L_RECORD_NUMBER]);
4
4
4 ! If flush requested then mark the output buffer and contine
4 ! at OUTPUT_FILTER
4 !
4 IF .SERVICE_STATUS EQL PSM$_FLUSH
4 THEN
5     BEGIN
5     $BBLOCK [.SCB[PSM$A_IOB],IOB_V_FLUSH_PENDING] = 1;
5     SCB[PSM$B_STATE] = OUTPUT_FILTER;
4     END;
4
3 END;
```

```
3 [INPUT_FILTER]:
4 BEGIN
4
4 ! Locate the input filter
4 !
4 BIND FILTER = PSM$SRV[PSM$K_INPUT_FILTER,0,0,0,0] : $BBLOCK;
4
4
4 ! If no filter then go to filter completion
4 !
4 IF .FILTER[SRV_A_SERVICE] EQL 0
4 THEN
4     LEAVE CASE_STATEMENT;
4
4
4 ! Copy the descriptor (any class) and initialize the old one (dynamic)
4 !
4 COPY_QUAD_ (SCB[PSM$Q_USER_RECORD], SCB[PSM$Q_INPUT_RECORD]);
4 INIT_DYN_DESC_ (SCB[PSM$Q_USER_RECORD]);
4
4
4 ! Initiate the filter operation
4 !
4 SCB[PSM$L_SERVICE_STATUS] = BLISS (
4     .FILTER[SRV_A_SERVICE],                 !  - input filter service
4     SCB,                                    !  - SCB address by reference
4     SCB[PSM$R_USER_CONTEXT_AREA],           !  - user context area
4     UPLIT (PSM$K_FORMAT),                   !  - FORMAT function
4     SCB[PSM$Q_INPUT_RECORD],                !  - input record descriptor
4     SCB[PSM$L_CARCON],                      !  - input carriage control
4     SCB[PSM$Q_USER_RECORD],                 !  - quadword to receive descriptor
4     SCB[PSM$L_CARCON]);                     !  - output carriage control
4
3 END;
```

```
3 [INPUT_FILTER_COMPLETION]:
4 BEGIN
4
4 ! Initialize the input record descriptor (static)
4 !
4 STR$ANALYZE_SDESC_R1 (
4     SCB[PSM$Q_USER_RECORD]                          ! Input record descriptor
4     ;
4     VECTOR [SCB[PSM$Q_INPUT_RECORD],0],             ! R0 -> size
4     VECTOR [SCB[PSM$Q_INPUT_RECORD],1]);            ! R1 -> address
4
4
4 ! If the first byte of the record was used for carriage control
4 ! (eg. FORTRAN) then remove it from the record descriptor
4 !
4 IF CARRIAGE_CONTROL (.SCB) EQL PSM$K_FIRST_CHAR_USED
4 THEN
5     BEGIN
5     DECREMENT_ (SCB_SIZE_ (INPUT_RECORD));
5     INCREMENT_ (SCB_ADDR_ (INPUT_RECORD));
5     END;
4
4
4 ! If this is an offset read (that is, one that is to begin in the
4 ! middle of a record) then adjust the record descriptor by the offset
4 ! value from the checkpoint.
4 !
4 IF TESTBITSC (SCB[PSM$V_READ_OFFSET])
4 THEN
5     BEGIN
5     BIND CHECKPOINT = .SCB[PSM$A_CHECKPOINT] : $BBLOCK;
5     SCB_SIZE_ (INPUT_RECORD) = .SCB_SIZE_ (INPUT_RECORD)
5         - .CHECKPOINT[SMBMSG$W_OFFSET];
5     SCB_ADDR_ (INPUT_RECORD) = .SCB_ADDR_ (INPUT_RECORD)
5         + .CHECKPOINT[SMBMSG$W_OFFSET];
5     SCB[PSM$L_CARCON] = .CHECKPOINT[SMBMSG$L_CARCON];
5     SCB[PSM$L_RECORD_NUMBER] = .CHECKPOINT[SMBMSG$L_RECORD_NUMBER];
4     END;
4
4
3 END;
```

```
3 [FORMAT]:
4 BEGIN
4
4 ! Locate the main format routine
4 !
4 BIND FILTER  = PSM$SRV[PSM$K_MAIN_FORMAT,0,0,0,0] : $BBLOCK;
4
4
4 ! Initiate the FORMAT function
4 !
4 SCB[PSM$L_SERVICE_STATUS] = BLISS (
4       .FILTER[SRV_A_SERVICE],                  !  - format service
4       SCB,                                     !  - SCB address by reference
4       SCB[PSM$R_USER_CONTEXT_AREA],            !  - user context area
4       UPLIT (PSM$K_FORMAT),                    !  - FORMAT function
4       SCB[PSM$Q_INPUT_RECORD],                 !  - input record descriptor
4       SCB[PSM$L_CARCON],                       !  - input carriage control
4       SCB[PSM$Q_OUTPUT_BUFFER],                !  - output buffer descriptor
4       0);                                      !  - unused function argument
4
3 END;
```

```
3 [FORMAT_COMPLETION]:
4 BEGIN
4
4 ! If succesfull then block multiple input records into a single
4 ! output buffer by continuing at READ.
4 !
4 IF .SERVICE_STATUS
4 THEN
5     BEGIN
5     SCB[PSM$B_STATE] = READ;
5     LEAVE CASE_STATEMENT;
4     END;
4
4
4 ! If starting an escape sequence then mark escape in progress.
4 ! Insure that there are at least two bytes remaining in the output
4 ! buffer to allow two-byte escape sequences to be assembled.
4 !
4 IF .SERVICE_STATUS EQL PSM$_ESCAPE
4 THEN
5     BEGIN
5     SCB[PSM$B_ESCAPE_STATE] = 0;
5     SCB[PSM$V_ESCAPE_IN_PROGRESS] = 1;
5     SCB[PSM$B_STATE] = FORMAT;
5
5     ! If there are at least two output bytes remaining then continue
5     ! at FORMAT, else write the buffer.
5     !
5     IF .SCB_SIZE_ (OUTPUT_BUFFER) GTRU 2
5     THEN
5         SCB[PSM$B_STATE] = FORMAT;
5     LEAVE CASE_STATEMENT;
4     END;
4
4
4 ! See if format service requesting suspension (OSC)
4 !
4 IF .SERVICE_STATUS EQL PSM$_SUSPEND
4 THEN
5     BEGIN
5     SUSPEND_SERVICE (.SCB);
5     SCB[PSM$B_STATE] = FIND_WORK;
5     LEAVE CASE_STATEMENT;
4     END;
4
4
4 ! If output buffer full then write it
4 !
4 IF .SERVICE_STATUS EQL PSM$_BUFFEROVF
4 THEN
4     LEAVE CASE_STATEMENT;
4
4
4 ! Must be a new page
4 !
4 IF .SERVICE_STATUS NEQ PSM$_NEWPAGE THEN CODEERR_ ;
4
```

```
4
4 ! New page -- save a checkpoint if 32 pages have passed or if
4 ! we are stopping on this page
4 !
4 IF (.SCB[PSM$L_PAGE] AND %B '11111') EQL 0
4 OR .SCB[PSM$L_PAGE] GEQU .SCB[PSM$L_STOP_PAGE]
4 THEN
4     SAVE_CHECKPOINT (.SCB);
4
4
4 ! If we are stopping on this page then flush the output stream
4 ! and reset the "new page" trigger
4 !
4 IF .SCB[PSM$L_PAGE] GEQU .SCB[PSM$L_STOP_PAGE]
4 THEN
5     BEGIN
5     $BBLOCK [.SCB[PSM$A_IOB],IOB_V_FLUSH_PENDING] = 1;
5     SCB[PSM$L_LINE] = 0;
5     LEAVE CASE_STATEMENT;
4     END;
4
4
4 ! Check for string search -- if the output buffer is not empty
4 ! then force a buffer write
4 !
4 IF .SCB[PSM$V_SEARCH_FOR_STRING]
4 THEN
5     BEGIN
5     BIND IOB = .SCB[PSM$A_IOB] : $BBLOCK;
5     IF .SCB_SIZE_ (OUTPUT_BUFFER) NEQ .DESC_SIZE_ (IOB[IOB_Q_BUFFER])
5     THEN
5         ! Reset the new page trigger and force a buffer write
5         !
6         BEGIN
6         SCB[PSM$L_LINE] = 0;
6         LEAVE CASE_STATEMENT;
5         END;
4     END;
4
4
4 ! Check for page headers and/or page setup
4 !
4 IF .PRINT_FLAG_ (PAGE_HEADER) THEN SERVICE_LIST_ (PAGE_HEADER) = 1;
4 IF .SCB_SIZE_ (PAGE_SETUP_MODULES) NEQ 0
4 OR .PSM$SRV[PSM$K_PAGE_SETUP, SRV_V_USER_SUPPLIED]
4 THEN
4     SERVICE_LIST_ (PAGE_SETUP) = 1;
4
4
4 ! If page headers or setup required then suspend current input service
4 ! and continue at FIND_WORK.
4 !
4 IF .SERVICE_LIST_ (PAGE_HEADER)
4 OR .SERVICE_LIST_ (PAGE_SETUP)
4 THEN
5     BEGIN
5     SUSPEND_SERVICE (.SCB);
```

```
; 1008           1940  5          SCB[PSM$B_STATE] = FIND_WORK;
; 1009           1941  5          LEAVE CASE_STATEMENT;
; 1010           1942  4          END;
; 1011           1943  4
; 1012           1944  4
; 1013           1945  4  ! If new page with no side effects then continue at FORMAT
; 1014           1946  4  ! else go to next state (output_filter)
; 1015           1947  4  !
; 1016           1948  4  IF NOT .SCB[PSM$V_SEARCH_FOR_STRING]
; 1017           1949  4  THEN
; 1018           1950  4          SCB[PSM$B_STATE] = FORMAT;
; 1019           1951  4
; 1020           1952  3  END;
```

```
3 [OUTPUT_FILTER]:
4 BEGIN
4
4 ! Locate the output filter service, the output block, and the output record
4 !
4 BIND FILTER = PSM$SRV[PSM$K_OUTPUT_FILTER,0,0,0,0] : $BBLOCK;
4 BIND IOB = .SCB[PSM$A_IOB] : $BBLOCK;
4 BIND IOBREC = IOB[IOB_Q_RECORD] : VECTOR;
4
4
4 ! Clear the old record descriptor (any class) and set it to
4 ! the size of the blocked record buffer (static)
4 !
4 CLEAR_STRING_ (IOBREC);
4 IOBREC[1] = .DESC_ADDR_ (IOB[IOB_Q_BUFFER]);
4 IOBREC[0] = .SCB_ADDR_ (OUTPUT_BUFFER) - .DESC_ADDR_ (IOB[IOB_Q_BUFFER]);
4 IF .IOBREC[0] GTRU .DESC_SIZE_ (IOB[IOB_Q_BUFFER]) THEN CODEERR_ ;
4
4
4 ! If no output filter then bypass service call
4 !
4 IF .FILTER[SRV_A_SERVICE] EQL 0
4 THEN
4     LEAVE CASE_STATEMENT;
4
4
4 ! Copy the output record decriptor (static) and reinitialize it (dynamic)
4 !
4 COPY_QUAD_ (IOBREC, SCB[PSM$Q_OUTPUT_BUFFER]);
4 INIT_DYN_DESC_ (IOBREC);
4
4
4 ! Call the output filter service
4 !
4 SCB[PSM$L_SERVICE_STATUS] = BLISS (
4     .FILTER[SRV_A_SERVICE],             ! - output filter service
4     SCB,                                ! - SCB address by reference
4     SCB[PSM$R_USER_CONTEXT_AREA],       ! - user context area
4     UPLIT (PSM$K_FORMAT),               ! - FORMAT function
4     SCB[PSM$Q_OUTPUT_BUFFER],           ! - input to filter
4     0,                                  ! - unused function argument
4     IOBREC,                             ! - output from filter
4     0);                                 ! - unused function argument
4
3 END;
```

```
3 [OUTPUT_FILTER_COMPLETION]:
4 BEGIN
4
4 ! Locate the OUTPUT block
4 !
4 BIND IOB = .SCB[PSM$A_IOB] : $BBLOCK;
4
4
4 ! Check for string search
4 !
4 IF .SCB[PSM$V_SEARCH_FOR_STRING]
4 THEN
4     IF SEARCH_FOR_STRING (.SCB, SCB[PSM$Q_SEARCH_STRING], IOB[IOB_Q_RECORD])
4     THEN
4
4         ! String found -- release the output buffer, set the start
4         ! page, and continue at RESUME
4         !
5         BEGIN
5         INSERT_TAIL_ (.SCB[PSM$A_IOB], SCB[PSM$Q_BUFFER_QUEUE]);
5         SCB[PSM$A_IOB] = 0;
5         SCB[PSM$L_START_PAGE] = .SCB[PSM$L_PAGE];
5
5         ! If sitting at top of page then we really want to restart at
5         ! the previous page
5         !
5         IF .SCB[PSM$L_LINE] LEQU 1
5         AND .SCB[PSM$L_COLUMN] LEQU 1
5         AND .SCB[PSM$L_PAGE] GTRU 1
5         THEN
6             DECREMENT_ (SCB[PSM$L_START_PAGE])
5         ELSE
5             ! Mid-page: force RESUME to reposition by fibbing about
5             ! current page
5             !
5             INCREMENT_ (SCB[PSM$L_PAGE]);
5         SCB[PSM$B_STATE] = RESUME;
5         LEAVE CASE_STATEMENT;
4         END;
4
3 END;
```

```
3 [WRITE]:
4 BEGIN
4
4 ! Locate the output block and the output service routine
4 !
4 BIND IOB = .SCB[PSM$A_IOB] : $BBLOCK;
4 BIND OUTPUT = PSM$SRV[PSM$K_OUTPUT,0,0,0,0] : $BBLOCK;
4
4
4 ! Establish the default function as WRITE
4 !
4 LOCAL FUNCTION : INITIAL (PSM$K_WRITE);
4
4
4 ! Check for /PASSALL or buffer marked passall (DCS's)
4 !
4 IF .PRINT_FLAG_ (PASSALL)
4 OR .IOB[IOB_V_PASSALL]
4 THEN
4     FUNCTION = PSM$K_WRITE_NOFORMAT;
4
4
4 ! Check for write suppression (searching)
4 !
4 IF .SCB[PSM$V_SUPPRESS_OUTPUT]
4 THEN
4     FUNCTION = PSM$K_WRITE_SUPPRESSED
4 ELSE
4     INCREMENT_ (ACC_DATA_ (QIO_PUTS));
4
4
4 ! Initiate the WRITE function
4 !
4 SCB[PSM$L_SERVICE_STATUS] = BLISS (
4     .OUTPUT[SRV_A_SERVICE],                 !  - write service
4     SCB[PSM$A_IOB],                         !  - IOB address by reference
4     SCB[PSM$R_USER_CONTEXT_AREA],           !  - user context area
4     FUNCTION,                               !  - WRITE or WRITE_SUPPRESSED function
4     IOB[IOB_Q_RECORD],                      !  - record desc
4     0);                                     !  - <not used>
4
4
4 ! Disconnect the IOB from the SCB
4 !
4 SCB[PSM$A_IOB] = 0;
4
4
4 ! Asynchronous?
4 !
4 IF .SCB[PSM$L_SERVICE_STATUS] EQL PSM$_PENDING
4 THEN
5     BEGIN
5
5     ! Yes: don't wait for completion unless we are flushing the output stream
5     ! Either way, PSM$REPORT will release the IOB
5     !
5     IF NOT .IOB[IOB_V_FLUSH_PENDING]
```

```
5       THEN
6           BEGIN
6           SCB[PSM$L_SERVICE_STATUS] = SS$_NORMAL;
6           SCB[PSM$B_STATE] = FORMAT;
5           END;
5       END
4   ELSE
5       BEGIN
5       ! Synchronous return; release the IOB
5       !
5       INSERT_TAIL_ (IOB[IOB_Q_QLINKS], SCB[PSM$Q_BUFFER_QUEUE]);
5
5       ! If successful, and not flushing, then continue at FORMAT
5       !
5       IF .SCB[PSM$L_SERVICE_STATUS] EQL SS$_NORMAL
5       AND NOT .IOB[IOB_V_FLUSH_PENDING]
5       THEN
5           SCB[PSM$B_STATE] = FORMAT;
4       END;
4
3   END;
```

```
3 [WRITE_COMPLETION]:
4 BEGIN
4
4 ! If the IO failed then the error has already been stored and task abort begun.
4 ! continue at READ.
4 !
4 IF NOT .SCB[PSM$L_SERVICE_STATUS]
4 THEN
5     BEGIN
5     SCB[PSM$B_STATE] = READ;
5     LEAVE CASE_STATEMENT;
4     END;
4
4
4 ! The write was successful -- we are here because the output stream
4 ! is being flushed for one of:
4 !
4 !         1) Last page reached (PRINT /PAGE=last)
4 !         2) Job controller requested pause (STOP /QUEUE)
4 !         3) A page search operation has completed
4 !         4) An alignment operation has completed (START /QUEUE /ALIGN=pages)
4 !         5) We are in sheet feed mode (DEFINE /FORM /SHEET_FEED)
4 !
4 ! Respond based on why we are flushing
4 !
4
4 ! If pausing then mark the stream pending and respond to the job controller
4 !
4 IF .SCB[PSM$L_REQUEST_RESPONSE] EQL SMBMSG$K_PAUSE_TASK
4 THEN
5     BEGIN
5     SMB$SEND_TO_JOBCTL (
5         SCB[PSM$L_STREAM_INDEX],          ! - stream number
5         SCB[PSM$L_REQUEST_RESPONSE]);     ! - request response
5     SCB[PSM$V_RESUME_WAIT] = 1;
5     SCB[PSM$L_SERVICE_STATUS] = PSM$_PENDING;
5     LEAVE CASE_STATEMENT;
4     END;
4
4
4 ! If searching for a string then continue formatting
4 !
  IF .SCB[PSM$V_SEARCH_FOR_STRING]
4 THEN
5     BEGIN
5     SCB[PSM$B_STATE] = FORMAT;
5     LEAVE CASE_STATEMENT;
4     END;
4
4
4 ! If searching for a page or aligning then go to next state (resume)
4 !
4 IF .SCB[PSM$V_ALIGN]
4 OR .SCB[PSM$V_SEARCH_FOR_PAGE]
4 THEN
4     LEAVE CASE_STATEMENT;
4
```

```
4
4 ! Sheet feeding?
4 !
4 IF .$BBLOCK [SCB[PSM$L_PRINT_CONTROL], SMBMSG$V_SHEET_FEED]
4 AND NOT .SCB[PSM$V_SUPPRESS_OUTPUT]
4 THEN
5     BEGIN
5     LOCAL DEVICE_STATUS;
5     DEVICE_STATUS = .SCB[PSM$L_DEVICE_STATUS] OR SMBMSG$M_PAUSE_TASK;
5     SMB$SEND_TO_JOBCTL (
5         SCB[PSM$L_STREAM_INDEX],          ! - stream number
5         UPLIT (SMBMSG$K_TASK_STATUS),     ! - request response
5         0,                                ! - no accounting
5         0,                                ! - no checkpoint
5         DEVICE_STATUS                     ! - device status (paused)
5         );
5     SCB[PSM$V_RESUME_WAIT] = 1;
5     SCB[PSM$L_SERVICE_STATUS] = PSM$_PENDING;
5     LEAVE CASE_STATEMENT;
4     END;
4
4 IF .SCB[PSM$L_SERVICE_LIST] EQL 0 THEN SCB[PSM$B_STATE] = STOP_TASK
4 ELSE
4     IF .ITEM_PRESENT_ (LAST_PAGE)
4     AND .SCB[PSM$L_PAGE] GTRU .SCB[PSM$L_LAST_PAGE]
4     THEN
4         SCB[PSM$B_STATE] = CLOSE;
4
3 END;
```

```
3 [CLOSE]:
4 BEGIN
4
4 ! Defend against an attempt to CLOSE a non-existent service
4 !
4 IF .SERVICE[SRV_A_SERVICE] EQL 0
4 THEN
4     LEAVE CASE_STATEMENT;
4
4
4 ! Initiate the CLOSE function for the current input routine
4 !
4 SCB[PSM$L_SERVICE_STATUS] = BLISS (
4     .SERVICE[SRV_A_SERVICE],            !  - current input service
4     SCB,                                !  - SCB address by reference
4     SCB[PSM$R_USER_CONTEXT_AREA],       !  - user context area
4     UPLIT (PSM$K_CLOSE),                !  - CLOSE function
4     0,                                  !  - <not used>
4     0);                                 !  - <not used>
4
3 END;
```

```
3 [CLOSE_COMPLETION]:
4 BEGIN
4
4 ! Mark the service closed
4 !
4 BITVECTOR [SCB[PSM$L_SERVICE_OPEN], .SCB[PSM$B_SERVICE_INDEX]] = 0;
4
4
4 ! If this was a forced EOF and input was nested then pass the
4 ! abort flag to the next service, else clear it
4 !
4 IF TESTBITSC (SCB[PSM$V_EOF])
4 THEN
4     IF .SCB[PSM$B_INPUT_DEPTH] NEQ 0
4     THEN
4         SCB[PSM$V_EOF] = 1;
4
3 END;
```

```
3 [STOP_TASK]:
4 BEGIN
4
4 ! A stream is "active" if its queue is started.  It is busy if it
4 ! is currently processing a task.
4 !
4 LOCAL
4         ACTIVE_STREAMS  : INITIAL (0),          ! number of active streams
4         BUSY_STREAMS    : INITIAL (0)           ! number of busy streams
4         ;
4
4 ! Clear any pending input service routines from the service list and
4 ! reset the busy and reset flags.
4 !
4 SCB[PSM$L_SERVICE_LIST] = 0;
4 SCB[PSM$V_BUSY] = 0;
4 SCB[PSM$V_RESET] = 0;
4
4 ! If the job controller did not request an abort then we respond
4 ! with the asynchronous TASK_COMPLETE message.  Otherwise we respond
4 ! with the current contents of REQUEST_RESPONSE which is presumably
4 ! STOP_TASK or RESET_TASK.
4 !
4 IF .SCB[PSM$L_REQUEST_RESPONSE] EQL SMBMSG$K_START_TASK
4 OR .SCB[PSM$L_REQUEST_RESPONSE] EQL SMBMSG$K_RESUME_TASK
4 THEN
4     SCB[PSM$L_REQUEST_RESPONSE] = SMBMSG$K_TASK_COMPLETE;
4
4
4 ! Notify the job controller
4 !
4 SMB$SEND_TO_JOBCTL (
4         SCB[PSM$L_STREAM_INDEX],        ! - stream number
4         SCB[PSM$L_REQUEST_RESPONSE],    ! - responding to ...
4         SCB[PSM$Q_ACCOUNTING_DATA],     ! - accounting data
4         0,                              ! - no checkpoint
4         SCB[PSM$L_DEVICE_STATUS],       ! - device status
4         SCB[PSM$T_CONDITION_AREA]       ! - errors if any
4         );
4
4 ! Now scan to see if there are any active or busy streams
4 !
4 INCR I TO PSM$K_MAXSTREAMS - 1
4 DO
5     BEGIN
5     BIND SCBPTR  = .PSM$GL_SCBVEC [.I] : $BBLOCK;
5     IF SCBPTR NEQ 0
5     THEN
6         BEGIN
6         IF .SCBPTR[PSM$V_ACTIVE]
6         THEN
6             INCREMENT_ (ACTIVE_STREAMS);
6         IF .SCBPTR[PSM$V_BUSY]
6         THEN
6             INCREMENT_ (BUSY_STREAMS);
6         END
4     END;
```

```
; 1374          2299  4
; 1375          2300  4
; 1376          2301  4 ! If no active streams then exit
; 1377          2302  4 !
; 1378          2303  4 IF .ACTIVE_STREAMS EQL 0
; 1379          2304  4 THEN
; 1380          2305  4     $EXIT (CODE = SS$_NORMAL OR STS$M_INHIB_MSG);
; 1381          2306  4
; 1382          2307  4
; 1383          2308  4 ! If no busy streams then purge the working set
; 1384          2309  4 !
; 1385          2310  4 IF .BUSY_STREAMS EQL 0
; 1386          2311  4 THEN
; 1387          2312  4     $PURGWS (INADR=UPLIT (0, %X '7FFFFFFF'));
; 1388          2313  4
; 1389          2314  4
; 1390          2315  3 END;
```

```
3 [IDLE]:
4 BEGIN
4
4 ! If a reset has occured then contine at STOP_TASK
4 !
4 IF .SCB[PSM$V_RESET]
4 THEN
4     SCB[PSM$B_STATE] = STOP_TASK
4 ELSE
4     RETURN;
4
3 END;
```

```
; 1405          2328  3
; 1406          2329  3 | Usual formatting conventions resume here
; 1407          2330  3 !
; 1408          2331  3         TES;       ! End of case table
; 1409          2332  2     END;           ! End of CASE_STATEMENT block
; 1410          2333  1 END;               ! End of PSM$FUNCTION_DISPATCH routine

                                                                         .TITLE  DISPATCH Print Symbiont - main dispatch routine
;                                                                                                                               s
                                                                         .IDENT  \V04-000\

                                                                         .PSECT  CODE,NOWRT,2

0F  12  0D  0C  0B  0A  09  08  07  06  05  04  03  02  01   00000 NEXT_STATE:
                                                                         .BYTE   1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, -     ;
                                              12  11  11  01   0000F                     13, 18, 15, 1, 17, 17, 18               ;
                                                               00013          .BLKB   1
                                                    00000002   00014          .LONG   2                                        ;
                                                    00000000G  00018 P.AAA:   .LONG   PSM$_EOF                                 ;
                                                    0001827A   0001C          .LONG   98938                                    ;
                                                    00000004   00020          .LONG   4                                        ;
                     00000000G 00000000G 00000000G  00000000G  00024 P.AAB:   .LONG   PSM$_BUFFEROVF, PSM$_NEWPAGE, -          ;
                                                                                      PSM$_ESCAPE, PSM$_SUSPEND                ;
                                                         00#   00034 EXPECTED_ERRORS:
                                                                              .BYTE   0[20]                                    ;
                                                    00000000'  00048          .ADDRESS P.AAA                                   ;
                                                         00#   0004C          .BYTE   0[12]                                    ;
                                                    00000000'  00058          .ADDRESS P.AAB                                   ;
                                                               0005C          .BLKB   36
                                                    00000007   00080 P.AAC:   .LONG   7                                        ;
                                                    00000008   00084 P.AAD:   .LONG   8                                        ;
                                                    00000004   00088 P.AAE:   .LONG   4                                        ;
                                                    00000005   0008C P.AAF:   .LONG   5                                        ;
                                                    00000003   00090 P.AAG:   .LONG   3                                        ;
                                                    00000003   00094 P.AAH:   .LONG   3                                        ;
                                                    00000003   00098 P.AAI:   .LONG   3                                        ;
                                                    00000009   0009C P.AAJ:   .LONG   9                                        ;
                                                    00000002   000A0 P.AAK:   .LONG   2                                        ;
                                         7FFFFFFF   00000000   000A4 P.AAL:   .LONG   0, 2147483647                            ;

                                                                              .EXTRN  BAS$EDIT, LBR$CLOSE
                                                                              .EXTRN  LBR$GET_RECORD, LBR$INI_CONTROL
                                                                              .EXTRN  LBR$LOOKUP_KEY, LBR$OPEN
                                                                              .EXTRN  LBR$RET_RMSSTV, LBR$SET_LOCATE
                                                                              .EXTRN  LIB$TRIM_FILESPEC
                                                                              .EXTRN  LIB$GET_VM, LIB$FREE_VM
                                                                              .EXTRN  STR$ANALYZE_SDESC
                                                                              .EXTRN  STR$ANALYZE_SDESC_R1
                                                                              .EXTRN  STR$APPEND, STR$CONCAT
                                                                              .EXTRN  STR$COPY_DX, STR$COPY_R
                                                                              .EXTRN  STR$FREE1_DX, STR$FREE1_DX_R4
                                                                              .EXTRN  STR$GET1_DX, STR$LEFT
                                                                              .EXTRN  STR$PREFIX, STR$RIGHT
                                                                              .EXTRN  PSM$S_HANGUP_DISPATCH_ENTRY
                                                                              .EXTRN  PSM$_BUFFEROVF, PSM$_EOF
                                                                              .EXTRN  PSM$_ESCAPE, PSM$_FLUSH
```

```
                                                              .EXTRN  PSM$_FUNNOTSUP, PSM$_INVITMCOD
                                                              .EXTRN  PSM$_INVVMSOSC, PSM$_MODNOTFND
                                                              .EXTRN  PSM$_NEWPAGE, PSM$_NOFILEID
                                                              .EXTRN  PSM$_OSCTOOLON, PSM$_PENDING
                                                              .EXTRN  PSM$_SUSPEND, PSM$_TOOMANYLEV
                                                              .EXTRN  SMB$_INVSTMNBR, SMB$_INVSTRLEV
                                                              .EXTRN  SMB$_NOMOREITEMS
                                                              .EXTRN  PSM$ALLOCATE_DSB
                                                              .EXTRN  PSM$ALLOCATE_IOB
                                                              .EXTRN  PSM$DEALLOCATE_DSB
                                                              .EXTRN  SMB$INITIALIZE, PSM$RECEIVE_MESSAGE_AST
                                                              .EXTRN  PSM$SCHEDULE_NON_AST
                                                              .EXTRN  SMB$SEND_TO_JOBCTL
                                                              .EXTRN  PSM$WAIT_FOR_NON_AST
                                                              .EXTRN  PSM$GL_SCBVEC, PSM$GL_MAXBUF
                                                              .EXTRN  PSM$GL_USER_CTX
                                                              .EXTRN  PSM$SRV, PSM$XLATE_ALIGN
                                                              .EXTRN  PSM$XLATE_8BIT, SYS$EXIT
                                                              .EXTRN  SYS$PURGWS

                                        0FFC 00000            .ENTRY  PSM$FUNCTION_DISPATCH, Save R2,R3,R4,R5,R6,-; 1122
                                                                      R7,R8,R9,R10,R11                             ;
                  5E              14  C2 00002                SUBL2   #20, SP                                      ;
                  52       04     AC  D0 00005 1$:            MOVL    SCB, R2                                      ; 1198
                  54     0220     C2  9E 00009                MOVAB   544(R2), R4                                  ;
       00000000G  8F              64  D1 0000E                CMPL    (R4), #PSM$_PENDING                          ;
                                  01  12 00015                BNEQ    2$                                           ;
                                      04 00017                RET                                                  ;
                  58     01AC     C2  9E 00018 2$:            MOVAB   428(R2), R8                                  ; 1209
                                  68  D5 0001D                TSTL    (R8)                                         ;
                                  0B  12 0001F                BNEQ    3$                                           ;
                                  52  DD 00021                PUSHL   R2                                           ; 1211
           0000V  CF              01  FB 00023                CALLS   #1, GET_BUFFER                               ;
                  01              50  E8 00028                BLBS    R0, 3$                                       ;
                                      04 0002B                RET                                                  ;
                  5B     027D     C2  9E 0002C 3$:            MOVAB   637(R2), R11                                 ; 1219
                  50              6B  9A 00031                MOVZBL  (R11), R0                                    ;
                  50              10  C4 00034                MULL2   #16, R0                                      ;
                  55 00000000G0040    9E 00037                MOVAB   PSM$SRV[R0], SERVICE                         ;
                  6E              64  D0 0003F                MOVL    (R4), SERVICE_STATUS                         ; 1220
                  64              01  D0 00042                MOVL    #1, (R4)                                     ; 1221
                  56     02A7     C2  9E 00045                MOVAB   679(R2), R6                                  ; 1226
                  53              66  9A 0004A                MOVZBL  (R6), CURRENT_STATE                          ;
                  66     FF02 CF43    90 0004D                MOVB    NEXT_STATE[CURRENT_STATE], (R6)              ; 1227
                  57              6E  D0 00053                MOVL    SERVICE_STATUS, R7                           ; 1232
                  33              57  E8 00056                BLBS    R7, 8$                                       ;
                  51     FF2A CF43    D0 00059                MOVL    EXPECTED_ERRORS[CURRENT_STATE], R1           ; 1235
                                  59  D4 0005F                CLRL    EXPECTED_ERROR                               ;
                                  51  D5 00061                TSTL    R1                                           ; 1242
                                  1B  13 00063                BEQL    7$                                           ;
  5A         FC   A1              01  C3 00065                SUBL3   #1, -4(R1), R10                              ; 1244
                                  50  D4 0006A                CLRL    ERROR_INDEX                                  ;
                                  0D  11 0006C                BRB     6$                                           ;
            6140                  57  D1 0006E 4$:            CMPL    R7, (R1)[ERROR_INDEX]                        ; 1246
                                  05  12 00072                BNEQ    5$                                           ;
                  59              01  D0 00074                MOVL    #1, EXPECTED_ERROR                           ; 1249
                                  07  11 00077                BRB     7$                                           ; 1248
```

```
                                      50  D6 00079 5$:    INCL    ERROR_INDEX                                    ; 1246
                        5A            50  D1 0007B 6$:    CMPL    ERROR_INDEX, R10                               ;
                                      EE  1B 0007E        BLEQU   4$                                             ;
                        09            59  E8 00080 7$:    BLBS    EXPECTED_ERROR, 8$                             ; 1255
                              0084    8F  BB 00083        PUSHR   #^M<R2,R7>                                     ; 1257
                   0000V  CF          02  FB 00087        CALLS   #2, PSM$STORE_ERRORS                           ;
              12          00          53  CF 0008C 8$:    CASEL   CURRENT_STATE, #0, #18                         ; 1263
   01FF       01DF        01C5      01AD     00090 9$:    .WORD   27$-9$,-                                       ;
   0394       0358        02D9      0282     00098                28$-9$,-                                       ;
   0557       04D2        0403      03DD     000A0                30$-9$,-                                       ;
   06B9       069F        0619      05A8     000A8                32$-9$,-                                       ;
              0026        074C      06D6     000B0                39$-9$,-                                       ;
                                                                  44$-9$,-                                       ;
                                                                  52$-9$,-                                       ;
                                                                  53$-9$,-                                       ;
                                                                  56$-9$,-                                       ;
                                                                  58$-9$,-                                       ;
                                                                  75$-9$,-                                       ;
                                                                  79$-9$,-                                       ;
                                                                  83$-9$,-                                       ;
                                                                  89$-9$,-                                       ;
                                                                  103$-9$,-                                      ;
                                                                  107$-9$,-                                      ;
                                                                  111$-9$,-                                      ;
                                                                  119$-9$,-                                      ;
                                                                  10$-9$                                         ;
                   0244   C2 00000000G    00  9E 000B6 10$:   MOVAB   PSM$XLATE_8BIT, 580(R2)                        ; 1287
                          53        10    A2  9E 000BF        MOVAB   16(R2), R3                                     ; 1288
                          63      7001    8F  AA 000C3        BICW2   #28673, (R3)                                   ; 1291
                                  0224    C2  D5 000C8        TSTL    548(R2)                                        ; 1296
                                          07  12 000CC        BNEQ    11$                                            ;
                   0224   C2      01EC    C2  D0 000CE        MOVL    492(R2), 548(R2)                               ;
                                          52  DD 000D5 11$:   PUSHL   R2                                             ; 1301
                   0000V  CF              01  FB 000D7        CALLS   #1, FIND_CHECKPOINT                            ;
                                          50  D5 000DC        TSTL    CHECKPOINT                                     ; 1302
                                          35  13 000DE        BEQL    12$                                            ;
                   0190   C2              50  D0 000E0        MOVL    CHECKPOINT, 400(R2)                            ; 1309
                     01   A3              02  88 000E5        BISB2   #2, 1(R3)                                      ; 1315
                   01EC   C2        08    A0  D0 000E9        MOVL    8(CHECKPOINT), 492(R2)                         ; 1316
                                  0260    C2  B4 000EF        CLRW    608(R2)                                        ; 1317
                     0C   AE              04  D0 000F3        MOVL    #4, KEY_DESC                                   ; 1322
                     10   AE        10    A0  9E 000F7        MOVAB   16(R0), KEY_DESC+4                             ; 1323
                                          7E  D4 000FC        CLRL    -(SP)                                          ; 1331
                                    10    AE  9F 000FE        PUSHAB  KEY_DESC                                       ;
                                  FECF    CF  9F 00101        PUSHAB  P.AAC                                          ; 1332
                                  02D0    C2  9F 00105        PUSHAB  720(R2)                                        ; 1331
                                    04    AC  9F 00109        PUSHAB  SCB                                            ; 1328
                     00   B5              05  FB 0010C        CALLS   #5, @0(SERVICE)                                ; 1331
                          64              50  D0 00110        MOVL    R0, (R4)                                       ;
                                          34  11 00113        BRB     13$                                            ; 1337
                          53        04    AC  D0 00115 12$:   MOVL    SCB, R3                                        ; 1348
                   01EC   C3      0224    C3  D1 00119        CMPL    548(R3), 492(R3)                               ;
                                          48  1E 00120        BGEQU   15$                                            ;
                   01EC   C3              01  D0 00122        MOVL    #1, 492(R3)                                    ; 1354
                                  026C    C3  D4 00127        CLRL    620(R3)                                        ; 1355
                                  0260    C3  B4 0012B        CLRW    608(R3)                                        ; 1356
                                  0278    C3  D4 0012F        CLRL    632(R3)                                        ; 1357
```

```
                                   7E    7C 00133          CLRQ     -(SP)                                  ; 1365
                         FE9F      CF    9F 00135          PUSHAB   P.AAD                                  ; 1366
                         02D0      C3    9F 00139          PUSHAB   720(R3)                                ; 1365
                           04      AC    9F 0013D          PUSHAB   SCB                                    ; 1362
              00    B5             05    FB 00140          CALLS    #5, @0(SERVICE)                        ; 1365
            0220    C3             50    D0 00144          MOVL     R0, 544(R3)                            ;
                    50      04     AC    D0 00149  13$:    MOVL     SCB, R0                                ; 1371
       00000000G    8F    0220     C0    D1 0014D          CMPL     544(R0), #PSM$_FUNNOTSUP               ;
                                   0F    12 00156          BNEQ     14$                                    ;
                                   01    DD 00158          PUSHL    #1                                     ; 1372
                     01061154      8F    DD 0015A          PUSHL    #17174868                              ;
       00000000G    00             02    FB 00160          CALLS    #2, LIB$STOP                           ;
                         FE9B      31 00167  14$:          BRW      1$                                     ; 1375
                    53      04     AC    D0 0016A  15$:    MOVL     SCB, R3                                ; 1381
                    55    01EC     C3    9E 0016E          MOVAB    492(R3), R5                            ;
                    65    0224     C3    D1 00173          CMPL     548(R3), (R5)                          ;
                                   0D    1B 00178          BLEQU    16$                                    ;
            0228    C3    0224     C3    D0 0017A          MOVL     548(R3), 552(R3)                       ; 1384
              11    A3             10    88 00181          BISB2    #16, 17(R3)                            ; 1385
                                   5C    11 00185          BRB      20$                                    ; 1386
                    52    0228     C3    9E 00187  16$:    MOVAB    552(R3), R2                            ; 1394
                    62             01    CE 0018C          MNEGL    #1, (R2)                               ;
                    54    01B0     C3    9E 0018F          MOVAB    432(R3), R4                            ; 1395
      06            64             1B    E1 00194          BBC      #27, (R4), 17$                         ;
      62    00B8    C3             01    C1 00198          ADDL3    #1, 184(R3), (R2)                      ; 1397
      48            64             32    E5 0019E  17$:    BBCC     #50, (R4), 21$                         ; 1402
                                   30    DD 001A2          PUSHL    #48                                    ; 1405
                          014C     C3    9F 001A4          PUSHAB   332(R3)                                ;
                          014C     C3    9F 001A8          PUSHAB   332(R3)                                ;
       00000000G    00             03    FB 001AC          CALLS    #3, BAS$EDIT                           ;
                    01    0213     C3    91 001B3          CMPB     531(R3), #1                            ; 1406
                                   11    1A 001B8          BGTRU    18$                                    ;
                    50    0210     C3    9E 001BA          MOVAB    528(R3), R0                            ;
                    60 020E0000    8F    D0 001BF          MOVL     #34471936, (R0)                        ;
                            04     A0    D4 001C6          CLRL     4(R0)                                  ;
                                   14    11 001C9          BRB      19$                                    ;
                    50             53    D0 001CB  18$:    MOVL     R3, R0                                 ;
                          0210     C0    B5 001CE          TSTW     528(R0)                                ;
                                   0B    13 001D2          BEQL     19$                                    ;
                          0210     C3    9F 001D4          PUSHAB   528(R3)                                ;
       00000000G    00             01    FB 001D8          CALLS    #1, STR$FREE1_DX                       ;
              11    A3             20    88 001DF  19$:    BISB2    #32, 17(R3)                            ; 1407
              11    A3      40     8F    88 001E3  20$:    BISB2    #64, 17(R3)                            ; 1408
                                   4B    11 001E8          BRB      25$                                    ; 1409
      19            64             04    E5 001EA  21$:    BBCC     #4, (R4), 23$                          ; 1416
              10    A3             01    88 001EE          BISB2    #1, 16(R3)                             ; 1419
                    09    0140     C3    E9 001F2          BLBC     320(R3), 22$                           ; 1420
            0244    C3 00000000G   00    9E 001F7          MOVAB    PSM$XLATE_ALIGN, 580(R3)               ; 1422
      62            65      2C     A3    C1 00200  22$:    ADDL3    44(R3), (R5), (R2)                     ; 1423
                                   2E    11 00205          BRB      25$                                    ; 1424
      04    0124    C3             05    E1 00207  23$:    BBC      #5, 292(R3), 24$                       ; 1435
      62            65             01    C1 0020D          ADDL3    #1, (R5), (R2)                         ; 1437
                          0144     C3    9F 00211  24$:    PUSHAB   324(R3)                                ; 1444
                          022C     C3    9F 00215          PUSHAB   556(R3)                                ; 1443
       00000000G    00             02    FB 00219          CALLS    #2, SMB$SEND_TO_JOBCTL                 ; 1444
      0F    0140    C3             01    E5 00220          BBCC     #1, 320(R3), 25$                       ; 1449
              11    A3             08    88 00226          BISB2    #8, 17(R3)                             ; 1452
```

```
           0220  C3  00000000G   8F D0 0022A          MOVL     #PSM$_PENDING, 544(R3)                 ; 1453
                                 05 11 00233          BRB      26$                                    ; 1449
           02A7  C3              08 90 00235  25$:    MOVB     #8, 679(R3)                            ; 1456
                               FDC8 31 0023A  26$:    BRW      1$                                     ; 1263
                          0144   C2 9F 0023D  27$:    PUSHAB   324(R2)                                ; 1467
                          022C   C2 9F 00241          PUSHAB   556(R2)                                ; 1466
      00000000G  00              02 FB 00245          CALLS    #2, SMB$SEND_TO_JOBCTL                 ; 1467
E8         0140  C2              01 E5 0024C          BBCC     #1, 320(R2), 26$                       ; 1472
                               0472 31 00252          BRW      93$                                    ; 1475
03           11  A2              02 E1 00255  28$:    BBC      #2, 17(R2), 29$                        ; 1486
                               0584 31 0025A          BRW      120$                                   ;
                                 52 DD 0025D  29$:    PUSHL    R2                                     ; 1492
          0000V  CF              01 FB 0025F          CALLS    #1, SCHEDULE_SERVICE                   ;
                 D3              50 E8 00264          BLBS     R0, 26$                                ;
           0124  C2              20 8A 00267          BICB2    #32, 292(R2)                           ; 1497
                               016C 31 0026C          BRW      50$                                    ; 1498
                 50              6B 9A 0026F  30$:    MOVZBL   (R11), R0                              ; 1508
03         021C  C2              50 E1 00272          BBC      R0, 540(R2), 31$                       ;
                               045A 31 00278          BRW      95$                                    ;
           01A8  C2              02 D0 0027B  31$:    MOVL     #2, 424(R2)                            ; 1517
                          01A8   C2 9F 00280          PUSHAB   424(R2)                                ; 1528
                          0098   C2 9F 00284          PUSHAB   152(R2)                                ; 1527
                          FD50   CF 9F 00288          PUSHAB   P.AAE                                  ; 1526
                               04AA 31 0028C          BRW      104$                                   ; 1525
                 03              57 E8 0028F  32$:    BLBS     R7, 33$                                ; 1537
                               02BF 31 00292          BRW      72$                                    ;
                 50              6B 9A 00295  33$:    MOVZBL   (R11), R0                              ; 1547
00         021C  C2              50 E2 00298          BBSS     R0, 540(R2), 34$                       ;
           027C  C2       01A8   C2 90 0029E  34$:    MOVB     424(R2), 636(R2)                       ; 1548
                          02A5   C2 95 002A5          TSTB     677(R2)                                ; 1553
                                 05 14 002A9          BGTR     35$                                    ;
           0228  C2              01 CE 002AB          MNEGL    #1, 552(R2)                            ; 1555
                 01         0C   A5 91 002B0  35$:    CMPB     12(SERVICE), #1                        ; 1561
                                 84 12 002B4          BNEQ     26$                                    ;
           0204  C2       0124   C2 D0 002B6          MOVL     292(R2), 516(R2)                       ; 1567
           01BC  C2       00BC   C2 D0 002BD          MOVL     188(R2), 444(R2)                       ; 1573
           0230  C2       0164   C2 D0 002C4          MOVL     356(R2), 560(R2)                       ; 1574
50         008C  C2       00BC   C2 C3 002CB          SUBL3    188(R2), 140(R2), R0                   ; 1578
                 50       0148   C2 C2 002D3          SUBL2    328(R2), R0                            ; 1579
                 08              50 D1 002D8          CMPL     R0, #8                                 ;
                                 05 1E 002DB          BGEQU    36$                                    ;
           0204  C2              10 8A 002DD          BICB2    #16, 516(R2)                           ; 1581
                          01B0   C2 95 002E2  36$:    TSTB     432(R2)                                ; 1588
                                 1C 18 002E6          BGEQ     37$                                    ;
                 53         40   A2 D0 002E8          MOVL     64(R2), R3                             ; 1592
                 01         01   A3 91 002EC          CMPB     1(R3), #1                              ; 1593
                                 1D 12 002F0          BNEQ     38$                                    ;
                            3C   A2 9F 002F2          PUSHAB   60(R2)                                 ; 1596
                                 52 DD 002F5          PUSHL    R2                                     ;
          0000V  CF              02 FB 002F7          CALLS    #2, ENQUEUE_CHECKPOINT                 ;
           0224  C2         08   A3 D0 002FC          MOVL     8(R3), 548(R2)                         ; 1597
                                 0B 11 00302          BRB      38$                                    ; 1593
                 06       01B2   C2 E9 00304  37$:    BLBC     434(R2), 38$                           ; 1603
           0224  C2         74   A2 D0 00309          MOVL     116(R2), 548(R2)                       ; 1605
                               00C9 31 0030F  38$:    BRW      50$                                    ; 1610
                 01       0273   C2 91 00312  39$:    CMPB     627(R2), #1                            ; 1619
                                 11 1A 00317          BGTRU    40$                                    ;
```

```
                           50      0270   C2   9E 00319           MOVAB   624(R2), R0                              ;
                           60 020E0000    8F   D0 0031E           MOVL    #34471936, (R0)                          ;
                                     04   A0   D4 00325           CLRL    4(R0)                                    ;
                                          14   11 00328           BRB     41$                                      ;
                           50             52   D0 0032A  40$:     MOVL    R2, R0                                   ;
                                   0270   C0   B5 0032D           TSTW    624(R0)                                  ;
                                          0B   13 00331           BEQL    41$                                      ;
                                   0270   C2   9F 00333           PUSHAB  624(R2)                                  ;
                  00000000G        00     01   FB 00337           CALLS   #1, STR$FREE1_DX                         ;
            14          10         A2     02   E0 0033E  41$:     BBS     #2, 16(R2), 42$                          ; 1624
                                   0268   C2   D4 00343           CLRL    616(R2)                                  ; 1629
                        10   A2     80    8F   88 00347           BISB2   #128, 16(R2)                             ; 1630
                                          65   D5 0034C           TSTL    (SERVICE)                                ; 1635
                                          0A   12 0034E           BNEQ    43$                                      ;
                           6E 00000000G   8F   D0 00350           MOVL    #PSM$_FUNNOTSUP, SERVICE_STATUS          ; 1638
                                        FCAB   31 00357  42$:     BRW     1$                                       ; 1639
                                   0268   C2   9F 0035A  43$:     PUSHAB  616(R2)                                  ; 1651
                                   0270   C2   9F 0035E           PUSHAB  624(R2)                                  ; 1650
                                   FC7A   CF   9F 00362           PUSHAB  P.AAF                                    ; 1649
                                        03D0   31 00366           BRW     104$                                     ; 1648
                           0E             57   E9 00369  44$:     BLBC    R7, 45$                                  ; 1659
            09          10   A2           02   E0 0036C           BBS     #2, 16(R2), 45$                          ; 1660
                  00000000G  8F           57   D1 00371           CMPL    R7, #PSM$_FUNNOTSUP                      ; 1661
                                          50   12 00378           BNEQ    49$                                      ;
                             66           0E   90 0037A  45$:     MOVB    #14, (R6)                                ; 1667
                  00000000G  8F           57   D1 0037D           CMPL    R7, #PSM$_EOF                            ; 1672
                                          09   13 00384           BEQL    46$                                      ;
                  0001827A   8F           57   D1 00386           CMPL    R7, #98938                               ;
                                          3B   12 0038D           BNEQ    49$                                      ;
            09          11   A2           05   E0 0038F  46$:     BBS     #5, 17(R2), 47$                          ; 1673
            04          11   A2           04   E0 00394           BBS     #4, 17(R2), 47$                          ;
                             2D      10   A2   E9 00399           BLBC    16(R2), 49$                              ; 1674
                             01      0C   A5   91 0039D  47$:     CMPB    12(SERVICE), #1                          ; 1680
                                          05   12 003A1           BNEQ    48$                                      ;
                        11   A2      50   8F   8A 003A3           BICB2   #80, 17(R2)                              ; 1684
                             03    0144   C2   D1 003A8  48$:     CMPL    324(R2), #3                              ; 1690
                                          A8   12 003AD           BNEQ    42$                                      ;
                                          5E   DD 003AF           PUSHL   SP                                       ; 1695
                                          7E   7C 003B1           CLRQ    -(SP)                                    ;
                                          7E   D4 003B3           CLRL    -(SP)                                    ;
                                   0144   C2   9F 003B5           PUSHAB  324(R2)                                  ;
                                   022C   C2   9F 003B9           PUSHAB  556(R2)                                  ; 1694
                  00000000G  00           06   FB 003BD           CALLS   #6, SMB$SEND_TO_JOBCTL                   ; 1695
                             66           12   90 003C4           MOVB    #18, (R6)                                ; 1701
                                        02FD   31 003C7           BRW     93$                                      ; 1702
                                   0286   C2   D6 003CA  49$:     INCL    646(R2)                                  ; 1715
                                   026C   C2   D6 003CE           INCL    620(R2)                                  ; 1716
                  00000000G  8F           6E   D1 003D2           CMPL    SERVICE_STATUS, #PSM$_FLUSH              ; 1722
                                          0A   12 003D9           BNEQ    51$                                      ;
                             50           68   D0 003DB  50$:     MOVL    (R8), R0                                 ; 1725
                        2C   A0           02   88 003DE           BISB2   #2, 44(R0)                               ;
                             66           0A   90 003E2           MOVB    #10, (R6)                                ; 1726
                                        FC1D   31 003E5  51$:     BRW     1$                                       ; 1263
                           53 00000000G   00   D0 003E8  52$:     MOVL    FILTER, R3                               ; 1740
                                          F4   13 003EF           BEQL    51$                                      ;
                           51      0260   C2   9E 003F1           MOVAB   608(R2), R1                              ; 1747
                           50      0270   C2   9E 003F6           MOVAB   624(R2), R0                              ;
```

```
                61               60 7D 003FB       MOVQ    (R0), (R1)                    ;
                60    020E0000   8F D0 003FE       MOVL    #34471936, (R0)               ; 1748
                            04   A0 D4 00405       CLRL    4(R0)                         ;
                          0278   C2 9F 00408       PUSHAB  632(R2)                       ; 1761
                                 50 DD 0040C       PUSHL   R0                            ;
                          0278   C2 9F 0040E       PUSHAB  632(R2)                       ; 1759
                                 51 DD 00412       PUSHL   R1                            ; 1761
                          FBCC   CF 9F 00414       PUSHAB  P.AAG                         ; 1757
                          02D0   C2 9F 00418       PUSHAB  720(R2)                       ; 1756
                            04   AC 9F 0041C       PUSHAB  SCB                           ; 1753
                63               07 FB 0041F       CALLS   #7, (R3)                      ; 1761
                                 6C 11 00422       BRB     57$                           ;
                50        0270   C2 9E 00424 53$:  MOVAB   624(R2), R0                   ; 1770
                     00000000G   00 16 00429       JSB     STR$ANALYZE_SDESC_R1          ; 1773
                53        0260   C2 9E 0042F       MOVAB   608(R2), R3                   ; 1772
                63               50 7D 00434       MOVQ    R0, (R3)                      ;
                                 52 DD 00437       PUSHL   R2                            ; 1779
          0000V CF               01 FB 00439       CALLS   #1, CARRIAGE_CONTROL          ;
                03               50 D1 0043E       CMPL    R0, #3                        ;
                                 05 12 00441       BNEQ    54$                           ;
                                 63 B7 00443       DECW    (R3)                          ; 1782
                            04   A3 D6 00445       INCL    4(R3)                         ; 1783
9B           10 A2               09 E5 00448 54$:  BBCC    #9, 16(R2), 51$               ; 1791
                50        0190   C2 D0 0044D       MOVL    400(R2), R0                   ; 1794
                63          02   A0 A2 00452       SUBW2   2(R0), (R3)                   ; 1796
                51          02   A0 3C 00456       MOVZWL  2(R0), R1                     ; 1798
             04 A3               51 C0 0045A       ADDL2   R1, 4(R3)                     ;
           0278 C2          04   A0 D0 0045E       MOVL    4(R0), 632(R2)                ; 1799
           026C C2          0C   A0 D0 00464       MOVL    12(R0), 620(R2)               ; 1800
                               FB98 31 0046A 55$:  BRW     1$                            ; 1263
                50   00000000G   00 D0 0046D 56$:  MOVL    FILTER, R0                    ; 1816
                                 7E D4 00474       CLRL    -(SP)                         ; 1822
                          01E0   C2 9F 00476       PUSHAB  480(R2)                       ;
                          0278   C2 9F 0047A       PUSHAB  632(R2)                       ; 1821
                          0260   C2 9F 0047E       PUSHAB  608(R2)                       ; 1820
                          FB62   CF 9F 00482       PUSHAB  P.AAH                         ; 1819
                          02D0   C2 9F 00486       PUSHAB  720(R2)                       ; 1818
                            04   AC 9F 0048A       PUSHAB  SCB                           ; 1815
                60               07 FB 0048D       CALLS   #7, (R0)                      ; 1822
                               02B1 31 00490 57$:  BRW     105$                          ;
                03               57 E9 00493 58$:  BLBC    R7, 59$                       ; 1832
                               0213 31 00496       BRW     90$                           ;
      00000000G 8F               57 D1 00499 59$:  CMPL    R7, #PSM$_ESCAPE              ; 1844
                                 15 12 004A0       BNEQ    60$                           ;
                          02A3   C2 94 004A2       CLRB    675(R2)                       ; 1847
             10 A2               08 88 004A6       BISB2   #8, 16(R2)                    ; 1848
                66               08 90 004AA       MOVB    #8, (R6)                      ; 1849
                02        01E0   C2 B1 004AD       CMPW    480(R2), #2                   ; 1854
                                 B6 18 004B2       BLEQU   55$                           ;
                               021E 31 004B4       BRW     95$                           ; 1856
      00000000G 8F               57 D1 004B7 60$:  CMPL    R7, #PSM$_SUSPEND             ; 1863
                                 03 12 004BE       BNEQ    61$                           ;
                               008A 31 004C0       BRW     71$                           ;
      00000000G 8F               57 D1 004C3 61$:  CMPL    R7, #PSM$_BUFFEROVF           ; 1874
                                 9E 13 004CA       BEQL    55$                           ;
      00000000G 8F               57 D1 004CC       CMPL    R7, #PSM$_NEWPAGE             ; 1881
                                 0F 13 004D3       BEQL    62$                           ;
```

```
                                                        01 DD 004D5       PUSHL   #1                                    ;
                                             01061154   8F DD 004D7       PUSHL   #17174868                             ;
                             00000000G   00             02 FB 004DD       CALLS   #2, LIB$STOP                          ;
                                         1F      01EC   C2 93 004E4 62$:  BITB    492(R2), #31                          ; 1887
                                                        09 13 004E9       BEQL    63$                                   ;
                                  0228   C2      01EC   C2 D1 004EB       CMPL    492(R2), 552(R2)                      ; 1888
                                                        07 1F 004F2       BLSSU   64$                                   ;
                                                        52 DD 004F4 63$:  PUSHL   R2                                    ; 1890
                                 0000V   CF             01 FB 004F6       CALLS   #1, SAVE_CHECKPOINT                   ;
                                  0228   C2      01EC   C2 D1 004FB 64$:  CMPL    492(R2), 552(R2)                      ; 1896
                                                        09 1F 00502       BLSSU   65$                                   ;
                                         50             68 D0 00504       MOVL    (R8), R0                              ; 1899
                                    2C   A0             02 88 00507       BISB2   #2, 44(R0)                            ;
                                                        10 11 0050B       BRB     66$                                   ; 1900
                          11        11   A2             05 E1 0050D 65$:  BBC     #5, 17(R2), 67$                       ; 1908
                                         50             68 D0 00512       MOVL    (R8), R0                              ; 1911
                                    1C   A0      01E0   C2 B1 00515       CMPW    480(R2), 28(R0)                       ; 1912
                                                        06 13 0051B       BEQL    67$                                   ;
                                                 01CB   C2 D4 0051D 66$:  CLRL    456(R2)                               ; 1917
                                                        34 11 00521       BRB     73$                                   ; 1918
                          05      0204   C2             01 E1 00523 67$:  BBC     #1, 516(R2), 68$                      ; 1925
                                  0218   C2             04 88 00529       BISB2   #4, 536(R2)                           ;
                                                 00DC   C2 B5 0052E 68$:  TSTW    220(R2)                               ; 1926
                                                        08 12 00532       BNEQ    69$                                   ;
                          05 00000000G   00             01 E1 00534       BBC     #1, PSM$SRV+24, 70$                   ; 1927
                                  0218   C2             02 88 0053C 69$:  BISB2   #2, 536(R2)                           ; 1929
                          06      0218   C2             02 E0 00541 70$:  BBS     #2, 536(R2), 71$                      ; 1935
                          0D      0218   C2             01 E1 00547       BBC     #1, 536(R2), 74$                      ; 1936
                                                        52 DD 0054D 71$:  PUSHL   R2                                    ; 1939
                                 0000V   CF             01 FB 0054F       CALLS   #1, SUSPEND_SERVICE                   ;
                                         66             01 90 00554 72$:  MOVB    #1, (R6)                              ; 1940
                                                      FAAB 31 00557 73$:  BRW     1$                                    ; 1941
                          F8        11   A2             05 E0 0055A 74$:  BBS     #5, 17(R2), 73$                       ; 1948
                                                      0173 31 0055F       BRW     95$                                   ; 1950
                          53             68             24 C1 00562 75$:  ADDL3   #36, (R8), R3                         ; 1960
                                         01        03   A3 91 00566       CMPB    3(R3), #1                             ; 1966
                                                        0C 1A 0056A       BGTRU   76$                                   ;
                                         63  020E0000   8F D0 0056C       MOVL    #34471936, (R3)                       ;
                                                   04   A3 D4 00573       CLRL    4(R3)                                 ;
                                                        0D 11 00576       BRB     77$                                   ;
                                                        63 B5 00578 76$:  TSTW    (R3)                                  ;
                                                        09 13 0057A       BEQL    77$                                   ;
                                                        53 DD 0057C       PUSHL   R3                                    ;
                             00000000G   00             01 FB 0057E       CALLS   #1, STR$FREE1_DX                      ;
                                         50             68 D0 00585 77$:  MOVL    (R8), R0                              ; 1967
                                    04   A3        20   A0 D0 00588       MOVL    32(R0), 4(R3)                         ;
                          63      01E4   C2        20   A0 C3 0058D       SUBL3   32(R0), 484(R2), (R3)                 ; 1968
            63        1C    A0                     10   00 ED 00594       CMPZV   #0, #16, 28(R0), (R3)                 ; 1969
                                                        0F 1E 0059A       BGEQU   78$                                   ;
                                                        01 DD 0059C       PUSHL   #1                                    ;
                                             01061154   8F DD 0059E       PUSHL   #17174868                             ;
                             00000000G   00             02 FB 005A4       CALLS   #2, LIB$STOP                          ;
                                         51 00000000G   00 D0 005AB 78$:  MOVL    FILTER, R1                            ; 1974
                                                        A3 13 005B2       BEQL    73$                                   ;
                                         50             52 D0 005B4       MOVL    R2, R0                                ; 1981
                                  01E0   C0             63 D0 005B7       MOVL    (R3), 480(R0)                         ;
                                  01E4   C2        04   A3 D0 005BC       MOVL    4(R3), 484(R2)                        ;
```

```
                63 020E0000   8F  D0 005C2          MOVL     #34471936, (R3)                     ; 1982
                         04   A3  D4 005C9          CLRL     4(R3)                               ;
                              7E  D4 005CC          CLRL     -(SP)                               ; 1992
                              53  DD 005CE          PUSHL    R3                                  ;
                              7E  D4 005D0          CLRL     -(SP)                               ;
                       01E0   C2  9F 005D2          PUSHAB   480(R2)                             ;
                       FA12   CF  9F 005D6          PUSHAB   P.AAI                               ; 1991
                       02D0   C2  9F 005DA          PUSHAB   720(R2)                             ; 1990
                         04   AC  9F 005DE          PUSHAB   SCB                                 ; 1987
                61            07  FB 005E1          CALLS    #7, (R1)                            ; 1992
                            015D  31 005E4          BRW      105$                                ;
4A          11  A2            05  E1 005E7 79$:     BBC      #5, 17(R2), 82$                     ; 2008
7E              68            24  C1 005EC          ADDL3    #36, (R8), -(SP)                    ; 2010
                       014C   C2  9F 005F0          PUSHAB   332(R2)                             ;
                              52  DD 005F4          PUSHL    R2                                  ;
         0000V  CF            03  FB 005F6          CALLS    #3, SEARCH_FOR_STRING               ;
                38            50  E9 005FB          BLBC     R0, 82$                             ;
          0178  D2       00   B8  0E 005FE          INSQUE   @0(R8), @376(R2)                    ; 2017
                50       04   AC  D0 00604          MOVL     SCB, R0                             ; 2018
                       01AC   C0  D4 00608          CLRL     428(R0)                             ;
                51     01EC   C0  9E 0060C          MOVAB    492(R0), R1                         ; 2019
          0224  C0            61  D0 00611          MOVL     (R1), 548(R0)                       ;
                01     01C8   C0  D1 00616          CMPL     456(R0), #1                         ; 2024
                              12  1A 0061B          BGTRU    80$                                 ;
                01     0194   C0  D1 0061D          CMPL     404(R0), #1                         ; 2025
                              0B  1A 00622          BGTRU    80$                                 ;
                01            61  D1 00624          CMPL     (R1), #1                            ; 2026
                              06  1B 00627          BLEQU    80$                                 ;
                       0224   C0  D7 00629          DECL     548(R0)                             ; 2028
                              02  11 0062D          BRB      81$                                 ; 2024
                              61  D6 0062F 80$:     INCL     (R1)                                ; 2033
          02A7  C0            12  90 00631 81$:     MOVB     #18, 679(R0)                        ; 2034
                              77  11 00636 82$:     BRB      91$                                 ; 2035
                53            68  D0 00638 83$:     MOVL     (R8), R3                            ; 2044
            04  AE            09  D0 0063B          MOVL     #9, FUNCTION                        ; 2045
05        0204  C2            03  E0 0063F          BBS      #3, 516(R2), 84$                    ; 2055
04          2C  A3            02  E1 00645          BBC      #2, 44(R3), 85$                     ; 2056
            04  AE            0A  D0 0064A 84$:     MOVL     #10, FUNCTION                       ; 2058
06          11  A2            06  E1 0064E 85$:     BBC      #6, 17(R2), 86$                     ; 2063
            04  AE            0B  D0 00653          MOVL     #11, FUNCTION                       ; 2065
                              04  11 00657          BRB      87$                                 ;
                       0282   C2  D6 00659 86$:     INCL     642(R2)                             ; 2067
                50 00000000G  00  D0 0065D 87$:     MOVL     OUTPUT, R0                          ; 2073
                              7E  D4 00664          CLRL     -(SP)                               ; 2077
                         24   A3  9F 00666          PUSHAB   36(R3)                              ;
                         0C   AE  9F 00669          PUSHAB   FUNCTION                            ; 2075
                       02D0   C2  9F 0066C          PUSHAB   720(R2)                             ;
                              58  DD 00670          PUSHL    R8                                  ; 2077
                60            05  FB 00672          CALLS    #5, (R0)                            ;
                64            50  D0 00675          MOVL     R0, (R4)                            ;
                              68  D4 00678          CLRL     (R8)                                ; 2083
     00000000G  8F            64  D1 0067A          CMPL     (R4), #PSM$_PENDING                 ; 2088
                              0A  12 00681          BNEQ     88$                                 ;
50          2C  A3            01  E0 00683          BBS      #1, 44(R3), 96$                     ; 2095
                64            01  D0 00688          MOVL     #1, (R4)                            ; 2098
                              48  11 0068B          BRB      95$                                 ; 2099
          0178  D2            63  0E 0068D 88$:     INSQUE   (R3), @376(R2)                      ; 2106
```

```
                              50        04   AC  D0 00692           MOVL    SCB, R0                                         ; 2110
                              01      0220   C0  D1 00696           CMPL    544(R0), #1                                     ;
                                             73  12 0069B           BNEQ    99$                                             ;
               6E       2C    A3             01  E0 0069D           BBS     #1, 44(R3), 99$                                 ; 2111
                      02A7    C0             08  90 006A2           MOVB    #8, 679(R0)                                     ; 2113
                                             67  11 006A7           BRB     99$                                             ; 1263
                              05             64  E8 006A9 89$:      BLBS    (R4), 92$                                       ; 2123
                              66             04  90 006AC 90$:      MOVB    #4, (R6)                                        ; 2126
                                             7C  11 006AF 91$:      BRB     102$                                            ; 2127
                              01      0144   C2  D1 006B1 92$:      CMPL    324(R2), #1                                     ; 2145
                                             15  12 006B6           BNEQ    94$                                             ;
                                      0144   C2  9F 006B8           PUSHAB  324(R2)                                         ; 2150
                                      022C   C2  9F 006BC           PUSHAB  556(R2)                                         ; 2149
                  00000000G   00             02  FB 006C0           CALLS   #2, SMB$SEND_TO_JOBCTL                          ; 2150
                        11    A2             08  88 006C7 93$:      BISB2   #8, 17(R2)                                      ; 2151
                                             3C  11 006CB           BRB     98$                                             ; 2152
                              53        10   A2  9E 006CD 94$:      MOVAB   16(R2), R3                                      ; 2159
               05             63             0D  E1 006D1           BBC     #13, (R3), 97$                                  ;
                              66             08  90 006D5 95$:      MOVB    #8, (R6)                                        ; 2162
                                             6D  11 006D8 96$:      BRB     106$                                            ; 2163
                              6A             63  E8 006DA 97$:      BLBS    (R3), 106$                                      ; 2169
               66             63             0C  E0 006DD           BBS     #12, (R3), 106$                                 ; 2170
               2B     0124    C2             05  E1 006E1           BBC     #5, 292(R2), 100$                               ; 2177
               27             63             0E  E0 006E7           BBS     #14, (R3), 100$                                 ; 2178
        08     AE       54    A2             02  C9 006EB           BISL3   #2, 84(R2), DEVICE_STATUS                       ; 2182
                                        08   AE  9F 006F1           PUSHAB  DEVICE_STATUS                                   ; 2184
                                             7E  7C 006F4           CLRQ    -(SP)                                           ;
                                      F8F6   CF  9F 006F6           PUSHAB  P.AAJ                                           ; 2185
                                      022C   C2  9F 006FA           PUSHAB  556(R2)                                         ; 2184
                  00000000G   00             05  FB 006FE           CALLS   #5, SMB$SEND_TO_JOBCTL                          ;
                        01    A3             08  88 00705           BISB2   #8, 1(R3)                                       ; 2190
                              64 00000000G   8F  D0 00709 98$:      MOVL    #PSM$_PENDING, (R4)                             ; 2191
                                             52  11 00710 99$:      BRB     110$                                            ; 2192
                                      0218   C2  D5 00712 100$:     TSTL    536(R2)                                         ; 2195
                                             03  12 00716           BNEQ    101$                                            ;
                                           00C6  31 00718           BRW     120$                                            ;
               43     01B3    C2             03  E1 0071B 101$:     BBC     #3, 435(R2), 110$                               ; 2197
                      00B8    C2      01EC   C2  D1 00721           CMPL    492(R2), 184(R2)                                ; 2198
                                             3A  1B 00728           BLEQU   110$                                            ;
                              66             0E  90 0072A           MOVB    #14, (R6)                                       ; 2200
                                             35  11 0072D 102$:     BRB     110$                                            ; 1263
                                             65  D5 0072F 103$:     TSTL    (SERVICE)                                       ; 2208
                                             31  13 00731           BEQL    110$                                            ;
                                             7E  7C 00733           CLRQ    -(SP)                                           ; 2218
                                      F8BB   CF  9F 00735           PUSHAB  P.AAK                                           ; 2219
                                      02D0   C2  9F 00739 104$:     PUSHAB  720(R2)                                         ; 2218
                                        04   AC  9F 0073D           PUSHAB  SCB                                             ; 2215
                        00    B5             05  FB 00740           CALLS   #5, @0(SERVICE)                                 ; 2218
                              64             50  D0 00744 105$:     MOVL    R0, (R4)                                        ;
                                             1B  11 00747 106$:     BRB     110$                                            ; 1263
                              50             6B  9A 00749 107$:     MOVZBL  (R11), R0                                       ; 2229
               00     021C    C2             50  E5 0074C           BBCC    R0, 540(R2), 108$                               ;
               03       10    A2             02  E4 00752 108$:     BBSC    #2, 16(R2), 109$                                ; 2235
                                           0080  31 00757           BRW     118$                                            ;
                                      02A5   C2  95 0075A 109$:     TSTB    677(R2)                                         ; 2237
                                             7A  13 0075E           BEQL    118$                                            ;
                        10    A2             04  88 00760           BISB2   #4, 16(R2)                                      ; 2239
```

```
                                                                 I 11
DISPATCH        Print Symbiont - main dispatch routines         16-Sep-1984 02:10:00    VAX-11 Bliss-32 V4.0-742          Page  48
V04-000         FUNCTION_DISPATCH - Main symbiont control loop  14-Sep-1984 12:55:07    [PRTSMB.SRC]DISPATCH.B32;1            (25)

                                  7E  11 00764 110$:   BRB     121$                                      ; 1263
                                  53  7C 00766 111$:   CLRQ    BUSY_STREAMS                              ; 2243
                            0218  C2  D4 00768         CLRL    536(R2)                                   ; 2256
                 10    A2   0402  8F  AA 0076C         BICW2   #1026, 16(R2)                             ; 2258
                       50   0144  C2  9E 00772         MOVAB   324(R2), R0                               ; 2265
                       05         60  D1 00777         CMPL    (R0), #5                                  ;
                                  05  13 0077A         BEQL    112$                                      ;
                       03         60  D1 0077C         CMPL    (R0), #3                                  ; 2266
                                  03  12 0077F         BNEQ    113$                                      ;
                       60         08  D0 00781 112$:   MOVL    #8, (R0)                                  ; 2268
                            028E  C2  9F 00784 113$:   PUSHAB  654(R2)                                   ; 2279
                              54  A2  9F 00788         PUSHAB  84(R2)                                    ; 2278
                                  7E  D4 0078B         CLRL    -(SP)                                     ; 2279
                              14  A2  9F 0078D         PUSHAB  20(R2)                                    ; 2276
                                  50  DD 00790         PUSHL   R0                                        ; 2279
                            022C  C2  9F 00792         PUSHAB  556(R2)                                   ; 2274
             00000000G  00        06  FB 00796         CALLS   #6, SMB$SEND_TO_JOBCTL                    ; 2279
                                  51  D4 0079D         CLRL    I                                         ; 2284
                       50 00000000G0041  D0 0079F 114$:   MOVL    PSM$GL_SCBVEC[I], R0                   ; 2287
                                  0D  13 007A7         BEQL    116$                                      ; 2288
                       02     0C  A0  E9 007A9         BLBC    12(R0), 115$                              ; 2291
                                  54  D6 007AD         INCL    ACTIVE_STREAMS                            ; 2293
         02      10    A0         01  E1 007AF 115$:   BBC     #1, 16(R0), 116$                          ; 2294
                                  53  D6 007B4         INCL    BUSY_STREAMS                              ; 2296
         E5            51         1F  F3 007B6 116$:   AOBLEQ  #31, I, 114$                              ; 2288
                                  54  D5 007BA         TSTL    ACTIVE_STREAMS                            ; 2303
                                  0D  12 007BC         BNEQ    117$                                      ;
                        10000001  8F  DD 007BE         PUSHL   #268435457                                ; 2305
             00000000G  00        01  FB 007C4         CALLS   #1, SYS$EXIT                              ;
                                  53  D5 007CB 117$:   TSTL    BUSY_STREAMS                              ; 2310
                                  15  12 007CD         BNEQ    121$                                      ;
                            F825  CF  9F 007CF         PUSHAB  P.AAL                                     ; 2312
             00000000G  00        01  FB 007D3         CALLS   #1, SYS$PURGWS                            ;
                                  08  11 007DA 118$:   BRB     121$                                      ; 1263
         06      11    A2         02  E1 007DC 119$:   BBC     #2, 17(R2), 122$                          ; 2321
                       66         10  90 007E1 120$:   MOVB    #16, (R6)                                 ; 2323
                                F81E  31 007E4 121$:   BRW     1$                                        ;
                                      04 007E7 122$:   RET                                               ; 2333

; Routine Size:  2024 bytes,    Routine Base:  CODE + 00AC
```

```
1 %SBTTL 'COMPLETE_SERVICE - record completion for async. function'
1 ! Functional Description:
1 !                 Records completion of an asynchronous service function
1 !                 (one that was originally completed with PSM$_PENDING)
1 !                 and records the completion status.
1 !
1 ! Formal Parameters:
1 !                 SMB_CONTEXT : address of a SCB or an IOB
1 !                 USER_STATUS : address of longword contain completion status
1 !
1 ! Implicit Inputs:
1 !                 none
1 !
1 ! Implicit Outputs:
1 !                 none
1 !
1 ! Returned Value:
1 !                 SS$_NORMAL
1 !
1 ! Side Effects:
1 !                 SCB updated with completions status and DISPATCH called
1 !                 to resume processing
1 !--
1 GLOBAL ROUTINE PSM$REPORT (
1         SMB_CONTEXT     : REF $LONGWORD,        ! SCB or IOB address
1         USER_STATUS     : REF $LONGWORD         ! Completion status
1         ) =
2 BEGIN
2
2 ! Setup parameter referencing values
2 !
2 PARAMETER_INDEX_ (SMB_CONTEXT, USER_STATUS);
2
2 LOCAL
2         SCB             : REF $BBLOCK;
2
2 ! Pick up the context value
2 !
2 SCB = .SMB_CONTEXT[];
2
2
2 ! If the structure type -- if SCB then we have an SCB, else
2 ! we have an IOB.
2 !
2 IF .SCB[PSM$B_TYPE] EQL PSM$K_STRUCTURE_SCB
2 THEN
3     BEGIN
3     ! SCB -- we are completing an INPUT function.  If not currently
3     ! pending then something is wrong.
3     !
3     IF .SCB[PSM$L_SERVICE_STATUS] NEQ PSM$_PENDING THEN CODEERR_ ;
3
3     ! Pick up completion status, default is normal.
3     !
3     SCB[PSM$L_SERVICE_STATUS] = SS$_NORMAL;
3     IF PARAMETER_PRESENT_ (USER_STATUS)
3     THEN
```

```
                SCB[PSM$L_SERVICE_STATUS] = .USER_STATUS[];

            ! Call function dispatch to resume processing
            !
            PSM$FUNCTION_DISPATCH (.SCB);
            END

    ELSE

            BEGIN
            ! We have an IOB -- we are completing an asycn. output request

            LOCAL IOB : REF $BBLOCK;
            LOCAL OUTPUT_STATUS : INITIAL (SS$_NORMAL);

            ! Locate the IOB, check its structure type, and locate the SCB
            !
            IOB = .SCB;
            IF .IOB[IOB_B_TYPE] NEQ PSM$K_STRUCTURE_IOB THEN CODEERR_ ;
            SCB = .IOB[IOB_A_CONTEXT];

            ! Pick up the output completion status if specified -- default is normal
            !
            IF PARAMETER_PRESENT_ (USER_STATUS) THEN OUTPUT_STATUS = .USER_STATUS[];

            ! If no errors ...
            !
            IF .OUTPUT_STATUS
            THEN
                BEGIN

                ! Update accounting
                !
                INCREMENT_ (SCB[PSM$L_OUTPUT_QIOS]);

                ! If we have a checkpoint associated with this output buffer or
                ! if we are marked as stalled ...
                !
                IF .IOB[IOB_V_CHECKPOINT_PENDING]
                OR .$BBLOCK[SCB[PSM$L_DEVICE_STATUS], SMBMSG$V_STALLED]
                THEN
                    BEGIN

                    ! Then prepare to notify the job controller
                    !
                    LOCAL CHECKPOINT : INITIAL (0);
                    LOCAL CKP_DESC : VECTOR [2] PRESET ([0]=0);
                    LOCAL REQUEST_RESPONSE : INITIAL (SMBMSG$K_TASK_STATUS);

                    ! Output completion indicates we are no longer stalled
                    !
                    $BBLOCK [SCB[PSM$L_DEVICE_STATUS],SMBMSG$V_STALLED] = 0;

                    ! If we are also pausing then set the request response
                    ! to PAUSE_TASK.  By default it is TASK_STATUS which indicates
                    ! an asynchronous (unexpected) message to the job controller.
                    !
```

```
5           IF .IOB[IOB_V_PAUSE_PENDING]
5           THEN
5               REQUEST_RESPONSE = .SCB[PSM$L_REQUEST_RESPONSE];
5
5
5           ! If a checkpoint is present setup a descriptor for it
5           !
5           IF .IOB[IOB_V_CHECKPOINT_PENDING]
5           THEN
6               BEGIN
6               CKP_DESC[0] = SMBMSG$S_CHECKPOINT_DATA;
6               CKP_DESC[1] = IOB[IOB_T_CHECKPOINT_DATA];
6               CHECKPOINT = CKP_DESC;
5               END;
5
5           ! Notify the job controller of one or more:
5           !
5           !   - we are not stalled
5           !   - we have paused
5           !   - here is a checkpoint update
5
5           SMB$SEND_TO_JOBCTL (
5               SCB[PSM$L_STREAM_INDEX],                ! - stream number
5               REQUEST_RESPONSE,                       ! - request response
5               0,                                      ! - no accounting
5               .CHECKPOINT,                            ! - checkpoint or 0
5               SCB[PSM$L_DEVICE_STATUS]                ! - device status
5               );
4           END;
4       END
3   ELSE
3
3       ! Store any errors other than cancel or abort
3       !
3       IF .OUTPUT_STATUS EQL SS$_CANCEL OR .OUTPUT_STATUS EQL SS$_ABORT
3       THEN
3           1
3       ELSE
3           PSM$STORE_ERRORS (.SCB, PSM$_WRITEERR, 1, SCB[PSM$Q_DEVICE_NAME],
3               .OUTPUT_STATUS);
3
3   ! If we are flushing the output stream (that is, suspending further
3   ! input/format operations until all pending output has been printed)
3   ! then update the service status in the SCB with the output status.
3   !
3   IF .IOB[IOB_V_FLUSH_PENDING]
3   THEN
3       SCB[PSM$L_SERVICE_STATUS] = .OUTPUT_STATUS;
3
3   ! Release the IOB
3   !
3   INSERT_TAIL_ (IOB[IOB_Q_QLINKS], SCB[PSM$Q_BUFFER_QUEUE]);
3
3   ! Call dispatch to resume processing
3   !
3   PSM$FUNCTION_DISPATCH (.SCB);
2   END;
```

```
; 1583          2505  2
; 1584          2506  2 SS$_NORMAL
; 1585          2507  2
; 1586          2508  1 END;



                                               003C 00000          .ENTRY  PSM$REPORT, Save R2,R3,R4,R5              ; 2357
                        55 00000000G   00    9E 00002          MOVAB   LIB$STOP, R5                               ;
                        5E             0C    C2 00009          SUBL2   #12, SP                                    ;
                        52       04    BC    D0 0000C          MOVL    @SMB_CONTEXT, SCB                          ; 2372
                        03       08    A2    91 00010          CMPB    8(SCB), #3                                 ; 2378
                                       2D    12 00014          BNEQ    3$                                         ;
                        53     0220    C2    9E 00016          MOVAB   544(SCB), R3                               ; 2384
              00000000G 8F             63    D1 0001B          CMPL    (R3), #PSM$_PENDING                        ;
                                       0B    13 00022          BEQL    1$                                         ;
                                       01    DD 00024          PUSHL   #1                                         ;
                          01061154     8F    DD 00026          PUSHL   #17174868                                  ;
                        65             02    FB 0002C          CALLS   #2, LIB$STOP                               ;
                        63             01    D0 0002F  1$:     MOVL    #1, (R3)                                   ; 2388
                        02             6C    91 00032          CMPB    (AP), #2                                   ; 2389
                                       09    1F 00035          BLSSU   2$                                         ;
                                 08    AC    D5 00037          TSTL    8(AP)                                      ;
                                       04    13 0003A          BEQL    2$                                         ;
                        63       08    BC    D0 0003C          MOVL    @USER_STATUS, (R3)                         ; 2391
                                     00A8    31 00040  2$:     BRW     12$                                        ; 2395
                        54             01    D0 00043  3$:     MOVL    #1, OUTPUT_STATUS                          ; 2400
                        53             52    D0 00046          MOVL    SCB, IOB                                   ; 2408
                        02       08    A3    91 00049          CMPB    8(IOB), #2                                 ; 2409
                                       0B    13 0004D          BEQL    4$                                         ;
                                       01    DD 0004F          PUSHL   #1                                         ;
                          01061154     8F    DD 00051          PUSHL   #17174868                                  ;
                        65             02    FB 00057          CALLS   #2, LIB$STOP                               ;
                        52       14    A3    D0 0005A  4$:     MOVL    20(IOB), SCB                               ; 2410
                        02             6C    91 0005E          CMPB    (AP), #2                                   ; 2414
                                       09    1F 00061          BLSSU   5$                                         ;
                                 08    AC    D5 00063          TSTL    8(AP)                                      ;
                                       04    13 00066          BEQL    5$                                         ;
                        54       08    BC    D0 00068          MOVL    @USER_STATUS, OUTPUT_STATUS                ;
                        48             54    E9 0006C  5$:     BLBC    OUTPUT_STATUS, 9$                          ; 2418
                               01E8    C2    D6 0006F          INCL    488(SCB)                                   ; 2424
                        05       2C    A3    E8 00073          BLBS    44(IOB), 6$                                ; 2429
           60       54  A2             04    E1 00077          BBC     #4, 84(SCB), 10$                           ; 2430
                        50                   D4 0007C  6$:     CLRL    CHECKPOINT                                 ; 2432
                                 04    AE    7C 0007E          CLRQ    CKP_DESC                                   ; 2437
                        6E             09    D0 00081          MOVL    #9, REQUEST_RESPONSE                       ;
                    54  A2             10    8A 00084          BICB2   #16, 84(SCB)                               ; 2442
           05       2C  A3             03    E1 00088          BBC     #3, 44(IOB), 7$                            ; 2448
                        6E     0144    C2    D0 0008D          MOVL    324(SCB), REQUEST_RESPONSE                 ; 2450
                        0D       2C    A3    E9 00092  7$:     BLBC    44(IOB), 8$                                ; 2455
                    04  AE             18    D0 00096          MOVL    #24, CKP_DESC                              ; 2458
                    08  AE       30    A3    9E 0009A          MOVAB   48(R3), CKP_DESC+4                         ; 2459
                        50       04    AE    9E 0009F          MOVAB   CKP_DESC, CHECKPOINT                       ; 2460
                                 54    A2    9F 000A3  8$:     PUSHAB  84(SCB)                                    ; 2474
                                       50    DD 000A6          PUSHL   CHECKPOINT                                 ;
```

```
                                         7E  D4 000A8         CLRL     -(SP)                                     ;
                                  0C     AE  9F 000AA         PUSHAB   REQUEST_RESPONSE                          ; 2470
                                022C     C2  9F 000AD         PUSHAB   556(SCB)                                  ;
                    00000000G  00        05  FB 000B1         CALLS    #5, SMB$SEND_TO_JOBCTL                    ; 2474
                                         22  11 000B8         BRB      10$                                       ; 2418
                    00000830   8F        54  D1 000BA 9$:     CMPL     OUTPUT_STATUS, #2096                      ; 2482
                                         19  13 000C1         BEQL     10$                                       ;
                               2C        54  D1 000C3         CMPL     OUTPUT_STATUS, #44                        ;
                                         14  13 000C6         BEQL     10$                                       ;
                                         54  DD 000C8         PUSHL    OUTPUT_STATUS                             ; 2487
                                  4C     A2  9F 000CA         PUSHAB   76(SCB)                                   ; 2486
                                         01  DD 000CD         PUSHL    #1                                        ;
                            010610D2     8F  DD 000CF         PUSHL    #17174738                                 ;
                                         52  DD 000D5         PUSHL    SCB                                       ;
                       0000V   CF        05  FB 000D7         CALLS    #5, PSM$STORE_ERRORS                      ;
                05        2C   A3        01  E1 000DC 10$:    BBC      #1, 44(IOB), 11$                          ; 2493
                        0220   C2        54  D0 000E1         MOVL     OUTPUT_STATUS, 544(SCB)                   ; 2495
                        0178   D2        63  0E 000E6 11$:    INSQUE   (IOB), @376(SCB)                          ; 2499
                                         52  DD 000EB 12$:    PUSHL    SCB                                       ; 2503
                        F726   CF        01  FB 000ED         CALLS    #1, PSM$FUNCTION_DISPATCH                 ;
                               50        01  D0 000F2         MOVL     #1, R0                                    ; 2508
                                             04 000F5         RET                                                ;

; Routine Size:  246 bytes,    Routine Base:  CODE + 0894
```

```
%SBTTL 'INCLUDE_MODULES - queue text modules for inclusion'
! Functional Description:
!               Adds the specified modules to the queue of modules
!               that are waiting to be included in the input stream
!
! Formal Parameters:
!               SMB_CONTEXT     : assumed to be the SCB address
!               MODULE_LIST     : descriptor of comma separate module list
!
! Implicit Inputs:
!               none
!
! Implicit Outputs:
!               none
!
! Returned Value:
!               none
!
! Side Effects:
!               The modules are appended to the module list
!--
GLOBAL ROUTINE PSM$INCLUDE_MODULES (
        SMB_CONTEXT     : REF $LONGWORD,        ! SCB address
        MODULE_LIST     : REF VECTOR            ! Module list descriptor
        ) =
BEGIN

LOCAL SCB : REF $BBLOCK;


! Locate the SCB
!
SCB = .SMB_CONTEXT[];


! Check for empty list
!
IF .DESC_SIZE_ (.MODULE_LIST) EQL 0 THEN RETURN SS$_NORMAL;


! If the pending list is non-empty then append a comma prior
! to new modules
!
IF .DESC_SIZE_ (SCB[PSM$Q_MODULE_LIST]) NEQ 0
THEN
    STR$APPEND (SCB[PSM$Q_MODULE_LIST], %ASCID ',');


! Append the new modules
!
STR$APPEND (SCB[PSM$Q_MODULE_LIST], .MODULE_LIST);

SS$_NORMAL

END;
```

```
                                        0098A           .BLKB   2
                    00  00  00  2C      0098C P.AAN:    .ASCII  \,\<0><0><0>                                    ;
                          010E0001      00990 P.AAM:    .LONG   17694721                                        ;
                          00000000'     00994           .ADDRESS P.AAN                                          ;


                                 000C   00000           .ENTRY  PSM$INCLUDE_MODULES, Save R2,R3                 ; 2530
             53 00000000G   00   9E     00002           MOVAB   STR$APPEND, R3                                  ;
             50         04  BC   D0     00009           MOVL    @SMB_CONTEXT, SCB                               ; 2541
                        08  BC   B5     0000D           TSTW    @MODULE_LIST                                    ; 2546
                                 19  13 00010           BEQL    2$                                              ;
             52       01CC  C0   9E     00012           MOVAB   460(SCB), R2                                    ; 2552
                                 62  B5 00017           TSTW    (R2)                                            ;
                                 08  13 00019           BEQL    1$                                              ;
                        DA  AF   9F     0001B           PUSHAB  P.AAM                                           ; 2554
                                 52  DD 0001E           PUSHL   R2                                              ;
             63                  02  FB 00020           CALLS   #2, STR$APPEND                                  ;
                        08  AC   DD     00023 1$:       PUSHL   MODULE_LIST                                     ; 2559
                                 52  DD 00026           PUSHL   R2                                              ;
             63                  02  FB 00028           CALLS   #2, STR$APPEND                                  ;
             50                  01  D0 0002B 2$:       MOVL    #1, R0                                          ; 2563
                                     04 0002E           RET                                                     ;

; Routine Size:  47 bytes,    Routine Base:  CODE + 0998
```

```
DIS
V04
```

```
1 %SBTTL 'PRINT_SYMBIONT - initialization/main entry point for print symbiont'
1 ! Functional Description:
1 !                 Initializes the print symbiont and begins processing
1 !
1 ! Formal Parameters:
1 !                 STREAMS :       Number of streams to allow (1-16)
1 !                 BUFLIM  :       Maximum output buffer size to allow
1 !                 USER_SIZE :     User work area size to allocate
1 !
1 ! Implicit Inputs:
1 !                 none
1 !
1 ! Implicit Outputs:
1 !                 none
1 !
1 ! Returned Value:
1 !                 none
1 !
1 ! Side Effects:
1 !                 Symbiont processing is initiated
1 !--
1 GLOBAL ROUTINE PSM$PRINT (
1         STREAMS         : REF $LONGWORD,
1         BUFLIM          : REF $WORD,
1         USER_SIZE       : REF $WORD
1         ) =
2 BEGIN
2
2 ! Setup for parameter referencing
2 !
2 PARAMETER_INDEX_ (STREAMS, BUFLIM, USER_SIZE);
2
2 BUILTIN FP;
2
2 LOCAL
2
2         ARG_DESC : $DYNAMIC_DESC,
2
2         ! Privileges needed by standard symbiont
2         !
2         PRIVILEGE_MASK: $BBLOCK[8] PRESET (
2             [PRV$V_ALLSPOOL]    = 1,
2             [PRV$V_LOG_IO]      = 1,
2             [PRV$V_PHY_IO]      = 1,
2             [PRV$V_READALL]     = 1,
2             [PRV$V_SHARE]       = 1),
2
2         MAXSTREAMS      : INITIAL (1)
2         ;
2
2 ! Create an item list for GETSYI call
2 !
2 BIND ITMLST = $ITMLST_UPLIT ((ITMCOD=SYI$_MAXBUF, BUFADR=PSM$GL_MAXBUF));
2
2
2 ! Establish the main signal handler
2 !
```

```
  2 .FP = HANDLER;
  2
  2
  2 ! Get the needed priv's
  2 !
  2 SIGNAL_IF_ERROR_ ($SETPRV (ENBFLG=1, PRVADR=PRIVILEGE_MASK));
  2
  2
  2 ! Get the value of the sysgen parameter for maximum buffer size
  2 !
  2 SIGNAL_IF_ERROR_ ($GETSYIW (ITMLST=ITMLST));
  2
  2
  2 ! Compute the maximum allowed buffer size as the smaller of the
  2 ! system limit and the user limit, less 100 to allow for $QIO overhead
  2 !
  2 PSM$GL_MAXBUF = .PSM$GL_MAXBUF - 100;
  2 IF PARAMETER_PRESENT_ (BUFLIM)
  2 THEN
  2     IF .PSM$GL_MAXBUF GTRU .BUFLIM[]
  2     THEN
  2         PSM$GL_MAXBUF = .BUFLIM[];
  2
  2
  2 ! Store the maximum streams value supplied by the user
  2 !
  2 IF PARAMETER_PRESENT_ (STREAMS)
  2 THEN
  2     MAXSTREAMS = .STREAMS[];
  2
  2
  2 ! Store the user context area size requested by the user
  2 !
  2 IF PARAMETER_PRESENT_ (USER_SIZE)
  2 THEN
  2     PSM$GL_USER_CTX  = .USER_SIZE[];
  2
  2
  2 ! Call the SMB$ facility to initialize symbiont environment and
  2 ! message interface to the job controller
  2 !
P 2 SIGNAL_IF_ERROR_ (SMB$INITIALIZE (
P 2     UPLIT (SMBMSG$K_STRUCTURE_LEVEL),
P 2     PSM$RECEIVE_MESSAGE_AST,
  2     MAXSTREAMS));
  2
  2
  2 ! Purge the working set
  2 !
  2 $PURGWS (INADR=UPLIT (0, %X '7FFFFFFF'));
  2
  2
  2 ! Loop forever at non-ast level, hibernating.  Nearly all symbiont activity
  2 ! occurs at ast-level, but a few functions occur at non-ast.  If woken from
  2 ! hibernate then look for non-ast work to do.
  2 !
  2 WHILE 1
```

```
; 1758          2678  2 DO
; 1759          2679  3     BEGIN
; 1760          2680  3     PSM$WAIT_FOR_NON_AST (ARG_DESC);
; 1761          2681  4         BEGIN
; 1762          2682  4
; 1763          2683  4         ! Argument list pointed to by arg_desc is a longword array of
; 1764          2684  4         ! the following values:
; 1765          2685  4         !
; 1766          2686  4         !       [0]         = SCB
; 1767          2687  4         !       [1]         = AST routine to activate after user routine
; 1768          2688  4         !       [2]         = AST parameter for AST routine
; 1769          2689  4         !       [3]         = User level routine
; 1770          2690  4         !       [4]         = User level argument count
; 1771          2691  4         !       [5]:[end]   = User level argument list
; 1772          2692  4         !
; 1773          2693  4         LOCAL SCB : REF $BBLOCK;
; 1774          2694  4         BIND ARG_VECTOR = .DESC_ADDR_ (ARG_DESC) : VECTOR;
; 1775          2695  4
; 1776          2696  4         SCB = .ARG_VECTOR [0];
; 1777          2697  4         SCB[PSM$L_NON_AST_STATUS] = CALLG (ARG_VECTOR[4], .ARG_VECTOR[3]);
; 1778          2698  4
; 1779          2699  4         IF .ARG_VECTOR[1] NEQ 0
; 1780          2700  4         THEN
; 1781        P 2701  4             SIGNAL_IF_ERROR_ ($DCLAST (ASTADR=.ARG_VECTOR[1],
; 1782          2702  4                 ASTPRM=.ARG_VECTOR[2]));
; 1783          2703  3         END;
; 1784          2704  2     END;
; 1785          2705  2
; 1786          2706  2 SS$_NORMAL
; 1787          2707  2
; 1788          2708  1 END;



                                    009C7            .BLKB    1
                      104F  0004    009C8 P.AAO:     .WORD    4, 4175                                  ;
                        00000000G   009CC            .ADDRESS PSM$GL_MAXBUF                            ;
                        00000000    009D0            .LONG    0                                        ;
                        00000000    009D4            .LONG    0                                        ;
                        00000001    009D8 P.AAP:     .LONG    1                                        ;
            7FFFFFFF    00000000    009DC P.AAQ:     .LONG    0, 2147483647                            ;

                                          ITMLST=                 P.AAO
                                                     .EXTRN   SYS$SETPRV, SYS$GETSYIW
                                                     .EXTRN   SYS$DCLAST

                                003C 00000           .ENTRY   PSM$PRINT, Save R2,R3,R4,R5              ; 2585
             55 00000000G   00   9E 00002           MOVAB    PSM$GL_MAXBUF, R5                        ;
             54 00000000G   00   9E 00009           MOVAB    LIB$SIGNAL, R4                           ;
             5E             0C   C2 00010           SUBL2    #12, SP                                  ;
      06     AE       020E  8F   B0 00013           MOVW     #526, ARG_DESC+2                         ; 2600
                        08  AE   D4 00019           CLRL     ARG_DESC+4                               ;
                  80400090  8F   DD 0001C           PUSHL    #-2143289200                             ; 2609
      04     AE             08   B0 00022           MOVW     #8, PRIVILEGE_MASK+4                     ;
                        06  AE   D4 00026           CLRL     PRIVILEGE_MASK+6                         ;
                            01   DD 00029           PUSHL    #1                                       ;
             6D      0000V  CF   9E 0002B           MOVAB    HANDLER, (FP)                            ; 2621
```

```
                                                                 G 12
DISPATCH        Print Symbiont - main dispatch routines          16-Sep-1984 02:10:00    VAX-11 Bliss-32 V4.0-742              Page  59
V04-000         PRINT_SYMBIONT - initialization/main entry poin 14-Sep-1984 12:55:07    [PRTSMB.SRC]DISPATCH.B32;1             (28)

                                                     7E  7C 00030         CLRQ    -(SP)                                       ; 2626
                                                0C   AE  9F 00032         PUSHAB  PRIVILEGE_MASK                              ;
                                                     01  DD 00035         PUSHL   #1                                          ;
                            00000000G 00             04  FB 00037         CALLS   #4, SYS$SETPRV                              ;
                                      52             50  D0 0003E         MOVL    R0, STATUS                                  ;
                                      05             52  E8 00041         BLBS    STATUS, 1$                                  ;
                                                     52  DD 00044         PUSHL   STATUS                                      ;
                                      64             01  FB 00046         CALLS   #1, LIB$SIGNAL                              ;
                                                     7E  7C 00049 1$:     CLRQ    -(SP)                                       ; 2631
                                                     7E  D4 0004B         CLRL    -(SP)                                       ;
                                                94   AF  9F 0004D         PUSHAB  ITMLST                                      ;
                                                     7E  7C 00050         CLRQ    -(SP)                                       ;
                                                     7E  D4 00052         CLRL    -(SP)                                       ;
                            00000000G 00             07  FB 00054         CALLS   #7, SYS$GETSYIW                             ;
                                      52             50  D0 0005B         MOVL    R0, STATUS                                  ;
                                      05             52  E8 0005E         BLBS    STATUS, 2$                                  ;
                                                     52  DD 00061         PUSHL   STATUS                                      ;
                                      64             01  FB 00063         CALLS   #1, LIB$SIGNAL                              ;
                                      65  00000064   8F  C2 00066 2$:     SUBL2   #100, PSM$GL_MAXBUF                         ; 2637
                                      02             6C  91 0006D         CMPB    (AP), #2                                    ; 2638
                                                     11  1F 00070         BLSSU   3$                                          ;
                                                08   AC  D5 00072         TSTL    8(AP)                                       ;
                                                     0C  13 00075         BEQL    3$                                          ;
           65       08   BC           10             00  ED 00077         CMPZV   #0, #16, @BUFLIM, PSM$GL_MAXBUF             ; 2640
                                                     04  1E 0007D         BGEQU   3$                                          ;
                                      65        08   BC  3C 0007F         MOVZWL  @BUFLIM, PSM$GL_MAXBUF                      ; 2642
                                                     6C  95 00083 3$:     TSTB    (AP)                                        ; 2647
                                                     09  13 00085         BEQL    4$                                          ;
                                                04   AC  D5 00087         TSTL    4(AP)                                       ;
                                                     04  13 0008A         BEQL    4$                                          ;
                                      6E        04   BC  D0 0008C         MOVL    @STREAMS, MAXSTREAMS                        ; 2649
                                      03             6C  91 00090 4$:     CMPB    (AP), #3                                    ; 2654
                                                     0D  1F 00093         BLSSU   5$                                          ;
                                                0C   AC  D5 00095         TSTL    12(AP)                                      ;
                                                     08  13 00098         BEQL    5$                                          ;
                            00000000G 00        0C   BC  3C 0009A         MOVZWL  @USER_SIZE, PSM$GL_USER_CTX                 ; 2656
                                                     5E  DD 000A2 5$:     PUSHL   SP                                          ; 2665
                                         00000000G   00  9F 000A4         PUSHAB  PSM$RECEIVE_MESSAGE_AST                     ;
                                              FF46   CF  9F 000AA         PUSHAB  P.AAP                                       ;
                            00000000G 00             03  FB 000AE         CALLS   #3, SMB$INITIALIZE                          ;
                                      52             50  D0 000B5         MOVL    R0, STATUS                                  ;
                                      05             52  E8 000B8         BLBS    STATUS, 6$                                  ;
                                                     52  DD 000BB         PUSHL   STATUS                                      ;
                                      64             01  FB 000BD         CALLS   #1, LIB$SIGNAL                              ;
                                              FF34   CF  9F 000C0 6$:     PUSHAB  P.AAQ                                       ; 2670
                            00000000G 00             01  FB 000C4         CALLS   #1, SYS$PURGWS                              ;
                                                0C   AE  9F 000CB 7$:     PUSHAB  ARG_DESC                                    ; 2680
                            00000000G 00             01  FB 000CE         CALLS   #1, PSM$WAIT_FOR_NON_AST                    ;
                                      52        10   AE  D0 000D5         MOVL    ARG_DESC+4, R2                              ; 2694
                                      53             62  D0 000D9         MOVL    (R2), SCB                                   ; 2696
                                  0C  B2        10   A2  FA 000DC         CALLG   16(R2), @12(R2)                             ; 2697
                                01DC  C3             50  D0 000E1         MOVL    R0, 476(SCB)                                ;
                                                04   A2  D5 000E6         TSTL    4(R2)                                       ; 2699
                                                     E0  13 000E9         BEQL    7$                                          ;
                                                     7E  D4 000EB         CLRL    -(SP)                                       ; 2702
                                      7E        04   A2  7D 000ED         MOVQ    4(R2), -(SP)                                ;
                            00000000G 00             03  FB 000F1         CALLS   #3, SYS$DCLAST                              ;
```

```
                                52          50  D0 000F8        MOVL    R0, STATUS                                  ;
                                CD          52  E8 000FB        BLBS    STATUS, 7$                                  ;
                                            52  DD 000FE        PUSHL   STATUS                                      ;
                                64          01  FB 00100        CALLS   #1, LIB$SIGNAL                              ;
                                            C6  11 00103        BRB     7$                                          ; 2677

; Routine Size:  261 bytes,     Routine Base:  CODE + 09E4
```

```
1 %SBTTL 'STORE_ERRORS - store errors reported by user in SCB'
1 ! Functional Description:
1 !                 Store the vector of condition codes in the call
1 !                 in the SCB.
1 !
1 ! Formal Parameters:
1 !                 SMB_CONTEXT     : assumed to be SCB address
1 !                 <8(AP)>         : begining of condition list
1 !
1 ! Implicit Inputs:
1 !                 none
1 !
1 ! Implicit Outputs:
1 !                 Error conditions and associcated text are stored
1 !
1 ! Returned Value:
1 !                 SS$_NORMAL
1 !
1 ! Side Effects:
1 !                 none
1 !--
1 GLOBAL ROUTINE PSM$STORE_ERRORS (
1         SMB_CONTEXT     : REF $LONGWORD
1         ) =
2 BEGIN
2
2 BUILTIN AP;
2 MAP     AP : REF VECTOR;
2
2 LOCAL
2         CONDITION,
2         ERRORS          : REF VECTOR,
2         INDEX           : INITIAL (0),
2         SCB             : REF $BBLOCK
2         ;
2
2 ! Locate the SCB and condition vector area
2 !
2 SCB = .SMB_CONTEXT[];
2 ERRORS = SCB[PSM$T_CONDITION_AREA];
2
2
2 ! If previous errors reported then ignore these
2 !
2 IF .ERRORS[0] NEQ 0 THEN RETURN SS$_NORMAL;
2
2
2 ! Expand the condition codes into a text message
2 !
2 EXPAND_CONDITION_VECTOR (.SCB, .AP[0] - 1, AP[2], SCB[PSM$Q_CONDITION_TEXT]);
2
2
2 ! Mark errrors to print
2 !
2 SERVICE_LIST_ (FILE_ERRORS) = 1;
2
2
```

```
; 1847          2766  2 ! Store the errors passing over FAO arguments
; 1848          2767  2 !
; 1849          2768  2 INCR I FROM 2 TO .AP[0]
; 1850          2769  2 DO
; 1851          2770  3     BEGIN
; 1852          2771  3     CONDITION = .AP [.I];
; 1853          2772  3     IF .CONDITION NEQ 0
; 1854          2773  3     THEN
; 1855          2774  4         BEGIN
; 1856          2775  4         INCREMENT_ (INDEX);
; 1857          2776  4         IF .INDEX GTRU PSM$S_CONDITION_AREA / 4  - 1
; 1858          2777  4         THEN
; 1859          2778  4             EXITLOOP;
; 1860          2779  4         INCREMENT_ (ERRORS[0]);
; 1861          2780  4         ERRORS[.INDEX] = .CONDITION;
; 1862          2781  3         END;
; 1863          2782  3
; 1864          2783  3     ! If this is neither an RMS nor a system message then
; 1865          2784  3     ! the low 16 bits of the next argument are an FAO count.
; 1866          2785  3     ! Skip the count argument longword, and the number of
; 1867          2786  3     ! additional longwords specified by the count.
; 1868          2787  3     !
; 1869          2788  3     IF .$BBLOCK [CONDITION,STS$V_FAC_NO] NEQ RMS$_FACILITY
; 1870          2789  3     AND .$BBLOCK [CONDITION,STS$V_FAC_NO] NEQ 0
; 1871          2790  3     AND .I LSSU .AP[0]
; 1872          2791  3     THEN
; 1873          2792  3         I = .I + .(AP[.I+1])<0,16,0> + 1;
; 1874          2793  2     END;
; 1875          2794  2
; 1876          2795  2
; 1877          2796  2 ! Any error initiates a task abort
; 1878          2797  2 !
; 1879          2798  2 ABORT_TASK (.SCB);
; 1880          2799  2
; 1881          2800  2 SS$_NORMAL
; 1882          2801  2
; 1883          2802  1 END;




                                       003C 00000          .ENTRY  PSM$STORE_ERRORS, Save R2,R3,R4,R5                 ; 2730
                                     55   D4 00002          CLRL    INDEX                                              ; 2733
                  52       04        BC   D0 00004          MOVL    @SMB_CONTEXT, SCB                                  ; 2747
                  53     028E        C2   9E 00008          MOVAB   654(R2), ERRORS                                    ; 2748
                                     63   D5 0000D          TSTL    (ERRORS)                                           ; 2753
                                     59   12 0000F          BNEQ    5$                                                 ;
                         0198        C2   9F 00011          PUSHAB  408(SCB)                                           ; 2758
                           08        AC   9F 00015          PUSHAB  8(AP)                                              ;
        7E        6C                 01   C3 00018          SUBL3   #1, (AP), -(SP)                                    ;
                                     52   DD 0001C          PUSHL   SCB                                                ;
             0000V  CF               04   FB 0001E          CALLS   #4, EXPAND_CONDITION_VECTOR                        ;
             021A   C2               04   88 00023          BISB2   #4, 538(SCB)                                       ; 2763
                    50               01   D0 00028          MOVL    #1, I                                              ; 2768
                                     32   11 0002B          BRB     3$                                                 ;
                    54             6C40   D0 0002D 1$:      MOVL    (AP)[I], CONDITION                                 ; 2771
```

```
                                          0D  13 00031         BEQL     2$                                      ; 2772
                                          55  D6 00033         INCL     INDEX                                   ; 2775
                                04        55  D1 00035         CMPL     INDEX, #4                               ; 2776
                                          29  1A 00038         BGTRU    4$                                      ;
                                          63  D6 0003A         INCL     (ERRORS)                                ; 2779
                              6345        54  D0 0003C         MOVL     CONDITION, (ERRORS)[INDEX]              ; 2780
        01           54         0C        10  ED 00040 2$:     CMPZV    #16, #12, CONDITION, #1                 ; 2788
                                          18  13 00045         BEQL     3$                                      ;
        00           54         0C        10  ED 00047         CMPZV    #16, #12, CONDITION, #0                 ; 2789
                                          11  13 0004C         BEQL     3$                                      ;
                                6C        50  D1 0004E         CMPL     I, (AP)                                 ; 2790
                                          0C  1E 00051         BGEQU    3$                                      ;
                                   04 AC40    DF 00053         PUSHAL   4(AP)[I]                                ; 2792
                                51        9E  3C 00057         MOVZWL   @(SP)+, R1                              ;
                                50   01 A140  9E 0005A         MOVAB    1(R1)[I], I                             ;
                     CA         50        6C  F3 0005F 3$:     AOBLEQ   (AP), I, 1$                             ; 2768
                                          52  DD 00063 4$:     PUSHL    SCB                                     ; 2798
                         0000V  CF        01  FB 00065         CALLS    #1, ABORT_TASK                          ;
                                50        01  D0 0006A 5$:     MOVL     #1, R0                                  ; 2802
                                              04 0006D         RET                                              ;

; Routine Size:  110 bytes,    Routine Base:  CODE + 0AE9
```

```
1 %SBTTL 'ABORT_TASK - aborts the current task'
1 ! Functional Description:
1 !             Causes the current task to be aborted by setting abort
1 !             flags and cancelling unneeded input services.
1 !
1 ! Formal Parameters:
1 !             SCB     : SCB address
1 !
1 ! Implicit Inputs:
1 !             none
1 !
1 ! Implicit Outputs:
1 !             none
1 !
1 ! Returned Value:
1 !             none
1 !
1 ! Side Effects:
1 !             The current task is cancelled.
1 !--
1 ROUTINE ABORT_TASK (
1         SCB     : REF $BBLOCK
1         )       : NOVALUE =
2 BEGIN
2
2
2 ! If the main input routine has been requested but not
2 ! yet called with open, and if the file is actually opened
2 ! as evidenced by FAB_VALID being set, then close the file
2 ! directly since the main path will not call with a CLOSE function
2 !
2 IF .SERVICE_LIST_ (MAIN_INPUT)
2 AND .SCB[PSM$V_FAB_VALID]
2 THEN
2     $CLOSE (FAB=.SCB[PSM$A_FAB]);
2
2
2 ! Cancel any pending main input (file printing) and file setup.
2 !
2 SERVICE_LIST_ (MAIN_INPUT) = 0;
2 SERVICE_LIST_ (FILE_SETUP) = 0;
2
2
2 ! Turn on file trailer, job trailer, and/or job reset if the job controller
2 ! indicated they should occur on a task abort
2 !
2 IF .SEPARATE_FLAG_ (FILE_TRAILER_ABORT) THEN SERVICE_LIST_ (FILE_TRAILER)  = 1;
2 IF .SEPARATE_FLAG_ (JOB_TRAILER_ABORT)  THEN SERVICE_LIST_ (JOB_TRAILER)   = 1;
2 IF .SEPARATE_FLAG_ (JOB_RESET_ABORT)    THEN SERVICE_LIST_ (JOB_RESET)     = 1;
2
2
2 ! Clear any pending input modules
2 !
2 CLEAR_STRING_ (SCB[PSM$Q_MODULE_LIST]);
2
2
2 ! Set the master EOF flag to force wind-down while poping the input
```

```
; 1942          2860  2 ! service routine stack
; 1943          2861  2 !
; 1944          2862  2 SCB[PSM$V_EOF] = 1;
; 1945          2863  2
; 1946          2864  1 END;


                                                                  .EXTRN  SYS$CLOSE

                                             000C 00000 ABORT_TASK:
                                                                  .WORD   Save R2,R3                             ; 2823
                             52       04  AC  D0 00002            MOVL    SCB, R2                                ; 2834
                             53     0218  C2  9E 00006            MOVAB   536(R2), R3                            ;
                             10       02  A3  E9 0000B            BLBC    2(R3), 1$                              ;
                 08      10  A2           04  E1 0000F            BBC     #4, 16(R2), 1$                         ; 2835
                                    0248  C2  DD 00014            PUSHL   584(R2)                                ; 2837
                   00000000G 00           01  FB 00018            CALLS   #1, SYS$CLOSE                          ;
                         01  A3     0110  8F  AA 0001F 1$:        BICW2   #272, 1(R3)                            ; 2842
                             50     0154  C2  9E 00025            MOVAB   340(R2), R0                            ; 2849
                 04          60           03  E1 0002A            BBC     #3, (R0), 2$                           ;
                         02  A3           08  88 0002E            BISB2   #8, 2(R3)                              ;
                 04          60           09  E1 00032 2$:        BBC     #9, (R0), 3$                           ; 2850
                         02  A3           20  88 00036            BISB2   #32, 2(R3)                             ;
                                          60  95 0003A 3$:        TSTB    (R0)                                   ; 2851
                                          04  18 0003C            BGEQ    4$                                     ;
                         02  A3           10  88 0003E            BISB2   #16, 2(R3)                             ;
                             01     01CF  C2  91 00042 4$:        CMPB    463(R2), #1                            ; 2856
                                          11  1A 00047            BGTRU   5$                                     ;
                             50     01CC  C2  9E 00049            MOVAB   460(R2), R0                            ;
                             60 020E0000  8F  D0 0004E            MOVL    #34471936, (R0)                        ;
                                      04  A0  D4 00055            CLRL    4(R0)                                  ;
                                          14  11 00058            BRB     6$                                     ;
                             50           52  D0 0005A 5$:        MOVL    R2, R0                                 ;
                                    01CC  C0  B5 0005D            TSTW    460(R0)                                ;
                                          0B  13 00061            BEQL    6$                                     ;
                                    01CC  C2  9F 00063            PUSHAB  460(R2)                                ;
                   00000000G 00           01  FB 00067            CALLS   #1, STR$FREE1_DX                       ;
                         10  A2           04  88 0006E 6$:        BISB2   #4, 16(R2)                             ; 2862
                                              04 00072            RET                                            ; 2864

; Routine Size:  115 bytes,    Routine Base:  CODE + 0B57
```

```
1 %SBTTL 'CARRIAGE_CONTROL - compute carriage control'
1 ! Functional Description:
1 !                 Computes carriage control for input records with the
1 !                 assistance of the EXEC's carriage control routine.
1 !
1 ! Formal Parameters:
1 !                 SCB     : SCB address
1 !
1 ! Implicit Inputs:
1 !                 Carriage control type, first byte of input record,
1 !                 record header, form feed flags
1 !
1 ! Implicit Outputs:
1 !                 PSM$L_CARCON established
1 !
1 ! Returned Value:
1 !                 none
1 !
1 ! Side Effects:
1 !                 none
1 !--
1 ROUTINE CARRIAGE_CONTROL (
1         SCB     : REF $BBLOCK
1         ) =
2 BEGIN
2
2 ! Define JSB linkage to EXEC routine
2 !
2 LINKAGE
2         CARRIAGE_LINKAGE = JSB (REGISTER=3):
2                                 PRESERVE (3)
2                                 NOTUSED (2,4,5,6,7,8,9,10,11);
2
2 EXTERNAL ROUTINE
2         EXE$CARRIAGE:   CARRIAGE_LINKAGE NOVALUE;
2
2
2 ! Case on the carriage control type for this input routine
2 !
2 CASE .SCB[PSM$B_CC_TYPE] FROM 1 TO PSM$K_CC_MAX - 1 OF
2
2     SET
2
2     [OUTRANGE]:
2         CODEERR_ ;
2
2
2     ! Internal -- all carriage control is explicity imbedded in
2     ! the data records
2     !
2
2     [PSM$K_CC_INTERNAL]:
2         SCB[PSM$L_CARCON] = 0;
2
2
2
2     ! Implied -- generate leading <CR> and trailing <LF> for most
```

```
	! records with special handling for the first record from the
	! service and for form feeds in the first byte of a record.
	!

	[PSM$K_CC_IMPLIED]:
	    BEGIN

	    ! Default carriage control
	    !
	    SCB[PSM$L_CARCON] = PSM$K_LF_CR;


	    ! Clear leading carriage control for first record from service
	    !
	    IF .SCB[PSM$V_FIRST_RECORD]
	    THEN
		SCB[PSM$B_PREFIX_COUNT] = 0;


	    ! Clear leading carriage control if last record was FF only
	    !
	    IF TESTBITSC (SCB[PSM$V_IMPLICIT_FORMFEED])
	    THEN
		SCB[PSM$B_PREFIX_COUNT] = 0;


	    ! Check for form feed in first byte of record
	    !
	    IF .SCB_SIZE_ (INPUT_RECORD) GTRU 0
	    THEN
		IF CH$RCHAR (.SCB_ADDR_ (INPUT_RECORD)) EQL PSM$K_CHAR_FF
		THEN
		    BEGIN

		    ! First byte is form feed -- clear leading carriage control
		    !
		    SCB[PSM$B_PREFIX_COUNT] = 0;

		    ! One byte record -- clear trailing carriage control and set
		    ! implicit form feed flag to clear leading carriage control
		    ! for next record
		    !
		    IF .SCB_SIZE_ (INPUT_RECORD) EQL 1
		    THEN
			BEGIN
			SCB[PSM$B_POSTFIX_COUNT] = 0;
			SCB[PSM$V_IMPLICIT_FORMFEED] = 1;
			END;
		    END;
	    END;


	! Fortran -- first byte of the record defines carriage control
	!

	[PSM$K_CC_FORTRAN]:
	    IF .SCB_SIZE_ (INPUT_RECORD) EQL 0
```

```
; 2062          2979  2               THEN
; 2063          2980  2                   SCB[PSM$L_CARCON] = PSM$K_LF_CR
; 2064          2981  2               ELSE
; 2065          2982  3                   BEGIN
; 2066          2983  3                   SCB[PSM$L_CARCON] = CH$RCHAR (.SCB_ADDR_ (INPUT_RECORD));
; 2067          2984  3                   EXE$CARRIAGE (SCB[PSM$L_CARCON] - $BYTEOFFSET (IRP$B_CARCON));
; 2068          2985  3                   IF .SCB[PSM$B_PREFIX_CHAR] EQL 0
; 2069          2986  3                   THEN
; 2070          2987  3                       SCB[PSM$B_PREFIX_CHAR] = PSM$K_CHAR_LF;
; 2071          2988  3                   IF .SCB[PSM$B_POSTFIX_CHAR] EQL 0
; 2072          2989  3                   THEN
; 2073          2990  3                       SCB[PSM$B_POSTFIX_CHAR] = PSM$K_CHAR_LF;
; 2074          2991  3                   RETURN PSM$K_FIRST_CHAR_USED;
; 2075          2992  2                   END;
; 2076          2993  2
; 2077          2994  2
; 2078          2995  2
; 2079          2996  2       ! PRINT -- print file format (PRN).  Each record has a two byte
; 2080          2997  2       ! header that define carriage control.  DCL, for example, creates
; 2081          2998  2       ! PRN files.
; 2082          2999  2       !
; 2083          3000  2
; 2084          3001  2       [PSM$K_CC_PRINT]:
; 2085          3002  3           BEGIN
; 2086          3003  3           SCB[PSM$L_CARCON] = .SCB[PSM$L_RECORD_HEADER] ^ 16;
; 2087          3004  3           EXE$CARRIAGE (SCB[PSM$L_CARCON] - $BYTEOFFSET (IRP$B_CARCON));
; 2088          3005  3           IF .SCB[PSM$B_PREFIX_CHAR] EQL 0
; 2089          3006  3           THEN
; 2090          3007  3               SCB[PSM$B_PREFIX_CHAR] = PSM$K_CHAR_LF;
; 2091          3008  3           IF .SCB[PSM$B_POSTFIX_CHAR] EQL 0
; 2092          3009  3           THEN
; 2093          3010  3               SCB[PSM$B_POSTFIX_CHAR] = PSM$K_CHAR_LF
; 2094          3011  2           END;
; 2095          3012  2
; 2096          3013  2       TES;
; 2097          3014  2
; 2098          3015  2 RETURN SS$_NORMAL;
; 2099          3016  2
; 2100          3017  1 END;


                                                                        .EXTRN  EXE$CARRIAGE

                                                    001C 00000 CARRIAGE_CONTROL:
                                                                        .WORD   Save R2,R3,R4                              ; 2886
                               54 00000000G  00  9E 00002               MOVAB   EXE$CARRIAGE, R4                           ;
                               52        04  AC  D0 00009               MOVL    SCB, R2                                    ; 2904
                  03           01      027C  C2  8F 0000D               CASEB   636(R2), #1, #3                            ;
       0095       005A       001F           0019    00013 1$:           .WORD   2$-1$,-                                    ;
                                                                                3$-1$,-                                    ;
                                                                                6$-1$,-                                    ;
                                                                                11$-1$                                     ;
                                             01  DD 0001B               PUSHL   #1                                         ; 2908
                                  01061154   8F  DD 0001D               PUSHL   #17174868                                  ;
                       00000000G  00         02  FB 00023               CALLS   #2, LIB$STOP                               ;
                                             56  11 0002A               BRB     7$                                         ;
```

```
                                0278    C2  D4 0002C 2$:      CLRL    632(R2)                                          ; 2917
                                        50  11 00030          BRB     7$                                               ;
                        51      0278    C2  9E 00032 3$:      MOVAB   632(R2), R1                                      ; 2931
                        61  0D010A01    8F  D0 00037          MOVL    #218171905, (R1)                                 ;
         02      10     A2              05  E1 0003E          BBC     #5, 16(R2), 4$                                   ; 2936
                                        61  94 00043          CLRB    (R1)                                             ; 2938
         02      10     A2              06  E5 00045 4$:      BBCC    #6, 16(R2), 5$                                   ; 2943
                                        61  94 0004A          CLRB    (R1)                                             ; 2945
                        50      0260    C2  9E 0004C 5$:      MOVAB   608(R2), R0                                      ; 2950
                                        60  B5 00051          TSTW    (R0)                                             ;
                                        78  13 00053          BEQL    13$                                              ;
                        0C        04    B0  91 00055          CMPB    @4(R0), #12                                      ; 2952
                                        72  12 00059          BNEQ    13$                                              ;
                                        61  94 0005B          CLRB    (R1)                                             ; 2958
                        01              60  B1 0005D          CMPW    (R0), #1                                         ; 2964
                                        6B  12 00060          BNEQ    13$                                              ;
                                027A    C2  94 00062          CLRB    634(R2)                                          ; 2967
                 10     A2        40    8F  88 00066          BISB2   #64, 16(R2)                                      ; 2968
                                        60  11 0006B          BRB     13$                                              ; 2904
                        51      0278    C2  9E 0006D 6$:      MOVAB   632(R2), R1                                      ; 2980
                        50      0260    C2  9E 00072          MOVAB   608(R2), R0                                      ; 2978
                                        60  B5 00077          TSTW    (R0)                                             ;
                                        09  12 00079          BNEQ    8$                                               ;
                        61  0D010A01    8F  D0 0007B          MOVL    #218171905, (R1)                                 ; 2980
                                        49  11 00082 7$:      BRB     13$                                              ;
                        61        04    B0  9A 00084 8$:      MOVZBL  @4(R0), (R1)                                     ; 2983
                        53        C4    A1  9E 00088          MOVAB   -60(R1), R3                                      ; 2984
                                        64  16 0008C          JSB     EXE$CARRIAGE                                     ;
                                0279    C2  95 0008E          TSTB    633(R2)                                          ; 2985
                                        05  12 00092          BNEQ    9$                                               ;
                 0279   C2              0A  90 00094          MOVB    #10, 633(R2)                                     ; 2987
                                027B    C2  95 00099 9$:      TSTB    635(R2)                                          ; 2988
                                        05  12 0009D          BNEQ    10$                                              ;
                 027B   C2              0A  90 0009F          MOVB    #10, 635(R2)                                     ; 2990
                        50              03  D0 000A4 10$:     MOVL    #3, R0                                           ; 2991
                                            04 000A7          RET                                                      ;
  027B     C2    0268   C2              10  78 000A8 11$:     ASHL    #16, 616(R2), 632(R2)                            ; 3003
                        53      023C    C2  9E 000B0          MOVAB   572(R2), R3                                      ; 3004
                                        64  16 000B5          JSB     EXE$CARRIAGE                                     ;
                                0279    C2  95 000B7          TSTB    633(R2)                                          ; 3005
                                        05  12 000BB          BNEQ    12$                                              ;
                 0279   C2              0A  90 000BD          MOVB    #10, 633(R2)                                     ; 3007
                                027B    C2  95 000C2 12$:     TSTB    635(R2)                                          ; 3008
                                        05  12 000C6          BNEQ    13$                                              ;
                 027B   C2              0A  90 000C8          MOVB    #10, 635(R2)                                     ; 3010
                        50              01  D0 000CD 13$:     MOVL    #1, R0                                           ; 3015
                                            04 000D0          RET                                                      ; 3017

; Routine Size:  209 bytes,    Routine Base:  CODE + 0BCA
```

```
1 %SBTTL 'ENQUEUE_CHECKPOINT - add a checkpoint to the checkpoint queue'
1 ! Functional Description:
1 !                 This routine manages additions to the checkpoint queue.
1 !
1 ! Formal Parameters:
1 !                 SCB      : SCB address
1 !                 CKP_DESC: address of the checkpoint descriptor
1 !
1 ! Implicit Inputs:
1 !                 Checkpoint queue header
1 !
1 ! Implicit Outputs:
1 !                 none
1 !
1 ! Returned Value:
1 !                 none
1 !
1 ! Side Effects:
1 !                 The checkpoint is enqueued.  Memory may be allocated.
1 !                 The queue may be flushed.
1 !--
1 ROUTINE ENQUEUE_CHECKPOINT (
1         SCB             : REF $BBLOCK,
1         CKP_DESC        : REF VECTOR
1         ) : NOVALUE =
2 BEGIN
2
2 LOCAL
2         DSB             : REF $BBLOCK
2         ;
2
2
2 ! If the queue has reached its maximum depth then flush it by
2 ! discarding every other checkpoint
2 !
2 IF .SCB[PSM$B_CHECKPOINT_DEPTH] GTR PSM$K_CHECKPOINT_LIMIT
2 THEN
3     BEGIN
3     LOCAL FIRST_DSB     : REF $BBLOCK,
3           TOGGLE        : INITIAL (0);
3
3
3     ! Scan the queue by removing each checkpoint.  Every other
3     ! checkpoint is requeued.
3     !
3     FIRST_DSB = .FLINK_ (SCB[PSM$Q_CHECKPOINT_QUEUE]);
3     DO
4         BEGIN
4         REMOVE_HEAD_ (DSB, SCB[PSM$Q_CHECKPOINT_QUEUE]);
4         DSB = .DSB - $BYTEOFFSET (DSB_Q_QLINKS);
4         IF .TOGGLE
4         THEN
5             BEGIN
5             PSM$DEALLOCATE_DSB (.DSB);
5             DECREMENT_ (SCB[PSM$B_CHECKPOINT_DEPTH]);
5             IF .SCB[PSM$B_CHECKPOINT_DEPTH] LSS 0 THEN CODEERR_ ;
5             END
```

```
; 2159          3075 4              ELSE
; 2160          3076 4                  INSERT_TAIL_ (DSB[DSB_Q_QLINKS], SCB[PSM$Q_CHECKPOINT_QUEUE]);
; 2161          3077 4              INCREMENT_ (TOGGLE);
; 2162          3078 4              END
; 2163          3079 3          UNTIL
; 2164          3080 3              .FLINK_ (SCB[PSM$Q_CHECKPOINT_QUEUE]) EQL .FIRST_DSB;
; 2165          3081 2          END;
; 2166          3082 2
; 2167          3083 2
; 2168          3084 2  ! Allocate a dynamic string block, copy and enqueue the checkpoint
; 2169          3085 2  !
; 2170          3086 2  PSM$ALLOCATE_DSB (DSB);
; 2171          3087 2  COPY_DX_DX_ (CKP_DESC[0], DSB[DSB_Q_DESC]);
; 2172          3088 2  INSERT_TAIL_ (DSB[DSB_Q_QLINKS], SCB[PSM$Q_CHECKPOINT_QUEUE]);
; 2173          3089 2
; 2174          3090 2
; 2175          3091 2  ! Increment the checkpoint depth and check for coding error
; 2176          3092 2  !
; 2177          3093 2  INCREMENT_ (SCB[PSM$B_CHECKPOINT_DEPTH]);
; 2178          3094 2  IF .SCB[PSM$B_CHECKPOINT_DEPTH] LSS 0 THEN CODEERR_ ;   ! > 128
; 2179          3095 2
; 2180          3096 2  SS$_NORMAL
; 2181          3097 2
; 2182          3098 1  END;


                                        003C 00000 ENQUEUE_CHECKPOINT:
                                                          .WORD   Save R2,R3,R4,R5                       ; 3039
                      55 00000000G   00   9E 00002        MOVAB   LIB$STOP, R5                           ;
                      5E             04   C2 00009        SUBL2   #4, SP                                 ;
                      50          04 AC   D0 0000C        MOVL    SCB, R0                                ; 3053
                      14        02A2 C0   91 00010        CMPB    674(R0), #20                           ;
                                     54   15 00015        BLEQ    4$                                     ;
                                     53   D4 00017        CLRL    TOGGLE                                 ; 3055
                      54        017C C0   D0 00019        MOVL    380(R0), FIRST_DSB                     ; 3063
        52         04 AC 0000017C    8F   C1 0001E        ADDL3   #380, SCB, R2                          ; 3066
                      6E          00 B2   0F 00027 1$:    REMQUE  @0(R2), DSB                            ;
                      23             53   E9 0002B        BLBC    TOGGLE, 2$                             ; 3068
                                     6E   DD 0002E        PUSHL   DSB                                    ; 3071
             00000000G 00            01   FB 00030        CALLS   #1, PSM$DEALLOCATE_DSB                 ;
                      50          04 AC   D0 00037        MOVL    SCB, R0                                ; 3072
                      50        02A2 C0   9E 0003B        MOVAB   674(R0), R0                            ;
                                     60   97 00040        DECB    (R0)                                   ;
                                     17   18 00042        BGEQ    3$                                     ; 3073
                                     01   DD 00044        PUSHL   #1                                     ;
                         01061154    8F   DD 00046        PUSHL   #17174868                              ;
                      65             02   FB 0004C        CALLS   #2, LIB$STOP                           ;
                                     0A   11 0004F        BRB     3$                                     ; 3068
                      50          04 AC   D0 00051 2$:    MOVL    SCB, R0                                ; 3076
                0180  D0          00 BE   0E 00055        INSQUE  @DSB, @384(R0)                         ;
                                     53   D6 0005B 3$:    INCL    TOGGLE                                 ; 3077
        52         04 AC 0000017C    8F   C1 0005D        ADDL3   #380, SCB, R2                          ; 3080
                      54             62   D1 00066        CMPL    (R2), FIRST_DSB                        ;
                                     BC   12 00069        BNEQ    1$                                     ;
```

```
                                        5E  DD 0006B 4$:     PUSHL   SP                                          ; 3086
                     00000000G  00          01  FB 0006D         CALLS   #1, PSM$ALLOCATE_DSB                       ;
                                        08  AC  DD 00074         PUSHL   CKP_DESC                                    ; 3087
           7E           04  AE          08  C1 00077         ADDL3   #8, DSB, -(SP)                              ;
                     00000000G  00          02  FB 0007C         CALLS   #2, STR$COPY_DX                             ;
                                52          50  D0 00083         MOVL    R0, STATUS                                  ;
                                09          52  E8 00086         BLBS    STATUS, 5$                                  ;
                                            52  DD 00089         PUSHL   STATUS                                      ;
                     00000000G  00          01  FB 0008B         CALLS   #1, LIB$SIGNAL                              ;
                                50      04  AC  D0 00092 5$:     MOVL    SCB, R0                                     ; 3088
                      0180      D0      00  BE  0E 00096         INSQUE  @DSB, @384(R0)                              ;
                                50      04  AC  D0 0009C         MOVL    SCB, R0                                     ; 3093
                                50    02A2  C0  9E 000A0         MOVAB   674(R0), R0                                 ;
                                            60  96 000A5         INCB    (R0)                                        ;
                                            0B  18 000A7         BGEQ    6$                                          ; 3094
                                            01  DD 000A9         PUSHL   #1                                          ;
                                  01061154  8F  DD 000AB         PUSHL   #17174868                                   ;
                                65          02  FB 000B1         CALLS   #2, LIB$STOP                                ;
                                                04 000B4 6$:     RET                                                 ; 3098

; Routine Size:  181 bytes,    Routine Base:  CODE + 0C9B
; 2184            3099  1 %SBTTL 'EXPAND_CONDITION_VECTOR - expand condition codes to text'
; 2185            3100  1 ! Functional Description:
; 2186            3101  1 !                Expands a list of condition codes to concatenated
; 2187            3102  1 !                text messages.
; 2188            3103  1 !
; 2189            3104  1 ! Formal Parameters:
; 2190            3105  1 !                SCB      : SCB address
; 2191            3106  1 !                MSGCNT   : number of longwords in message vector
; 2192            3107  1 !                MSGVEC   : address of message vector
; 2193            3108  1 !                DESC     : address of descriptor to receive text
; 2194            3109  1 !
; 2195            3110  1 ! Implicit Inputs:
; 2196            3111  1 !                none
; 2197            3112  1 !
; 2198            3113  1 ! Implicit Outputs:
; 2199            3114  1 !                none
; 2200            3115  1 !
; 2201            3116  1 ! Returned Value:
; 2202            3117  1 !                none
; 2203            3118  1 !
; 2204            3119  1 ! Side Effects:
; 2205            3120  1 !                none
; 2206            3121  1 !--
; 2207            3122  1 GLOBAL ROUTINE EXPAND_CONDITION_VECTOR (
; 2208            3123  1         SCB      : REF $BBLOCK,
; 2209            3124  1         MSGCNT   ,
; 2210            3125  1         MSGVEC   : REF VECTOR,
; 2211            3126  1         DESC     : REF VECTOR            ! Dynamic descriptor to receive message
; 2212            3127  1         )        : NOVALUE =
; 2213            3128  2 BEGIN
; 2214            3129  2
; 2215            3130  2 BUILTIN AP;
; 2216            3131  2 LOCAL   TEMP : VECTOR [20];
; 2217            3132  2
; 2218            3133  2
; 2219            3134  2 ! Create a vector with message count in front, followed by messages
; 2220            3135  2 !
; 2221            3136  2 TEMP[0] = .MSGCNT;
; 2222            3137  2 CH$COPY (.MSGCNT * 4, .MSGVEC, 0, %ALLOCATION (TEMP) - 4, TEMP[1]);
; 2223            3138  2
; 2224            3139  2
; 2225            3140  2 ! Call $PUTMSG to look up text
; 2226            3141  2 !
; 2227          P 3142  2 SIGNAL_IF_ERROR_ ($PUTMSG (MSGVEC=TEMP, ACTRTN=PUTMSG_ACTION,
; 2228            3143  2         ACTPRM=.DESC));
; 2229            3144  2
; 2230            3145  1 END;


                                                                   .EXTRN   SYS$PUTMSG

                                                  003C 00000       .ENTRY   EXPAND_CONDITION_VECTOR, Save R2,R3,R4,R5     ; 3122
                            5E        B4   AE   9E 00002           MOVAB    -76(SP), SP                                   ;
                                      08   AC   DD 00006           PUSHL    MSGCNT                                        ; 3136
                   50       08   AC        02   78 00009           ASHL     #2, MSGCNT, R0                                ; 3137
   004C   8F       00       0C   BC        50   2C 0000E           MOVC5    R0, @MSGVEC, #0, #76, TEMP+4                  ;
```

```
                                     04   AE      00016                                                             ;
                                     10   AC  DD  00018          PUSHL   DESC                                       ; 3143
                                          7E  D4  0001B          CLRL    -(SP)                                      ;
                                  0000V   CF  9F  0001D          PUSHAB  PUTMSG_ACTION                              ;
                                     0C   AE  9F  00021          PUSHAB  TEMP                                       ;
                      00000000G  00       04  FB  00024          CALLS   #4, SYS$PUTMSG                             ;
                                 52       50  D0  0002B          MOVL    R0, STATUS                                 ;
                                 09       52  E8  0002E          BLBS    STATUS, 1$                                 ;
                                          52  DD  00031          PUSHL   STATUS                                     ;
                      00000000G  00       01  FB  00033          CALLS   #1, LIB$SIGNAL                             ;
                                              04  0003A 1$:      RET                                                ; 3145

; Routine Size:  59 bytes,    Routine Base:  CODE + 0D50
```

```
1 %SBTTL 'FIND_CHECKPOINT -- locate an appropriate checkpoint'
1 ! Functional Description:
1 !             Searches the checkpoint queue for the closest checkpoint
1 !             that preceeds the target page.
1 !
1 ! Formal Parameters:
1 !             SCB:    SCB ADDRESS
1 !
1 ! Implicit Inputs:
1 !             Checkpoint queue, start page
1 !
1 ! Implicit Outputs:
1 !             none
1 !
1 ! Returned Value:
1 !             Address of checkpoint or zero
1 !
1 ! Side Effects:
1 !             none
1 !--
1 ROUTINE FIND_CHECKPOINT (
1         SCB             : REF $BBLOCK
1         ) =
2 BEGIN
2
2 LOCAL
2         CLOSEST : REF $BBLOCK INITIAL (0),      ! Best checkpoint found
2         DSB     : REF $BBLOCK                   ! dynamic string block
2         ;
2
2
2 ! Initialize the queue pointer to the first item in the queue
2 !
2 DSB = .FLINK_ (SCB[PSM$Q_CHECKPOINT_QUEUE]);
2
2
2 ! Search the queue until we return to the queue header
2 !
2 UNTIL .DSB EQL SCB[PSM$Q_CHECKPOINT_QUEUE]
2 DO
3     BEGIN
3     BIND CKP = .DESC_ADDR_ (DSB[DSB_Q_DESC]) : $BBLOCK;
3
3     ! If this checkpoint preceeds the target page and is closer
3     ! than any other then save it
3     !
3     IF .CKP[SMBMSG$L_PAGE] LEQ .SCB[PSM$L_START_PAGE]
3     THEN
3         IF .CLOSEST EQL 0 THEN CLOSEST = CKP
3         ELSE
3             IF .CKP[SMBMSG$L_PAGE] GTRU .CLOSEST[SMBMSG$L_PAGE]
3             THEN
3                 CLOSEST = CKP;
3
3     ! Advance to the next queue entry
3     !
3     DSB = .FLINK_ (DSB[DSB_Q_QLINKS]);
```

```
; 2289          3203  2      END;
; 2290          3204  2
; 2291          3205  2
; 2292          3206  2 ! Return the address of the checkpoint if a useable one was found
; 2293          3207  2 !
; 2294          3208  2 IF .CLOSEST NEQ 0
; 2295          3209  2 THEN
; 2296          3210  2     ! If current page greater than target page,
; 2297          3211  2     ! or current page less than checkpoint page
; 2298          3212  2     !
; 2299          3213  2     IF .SCB[PSM$L_PAGE] GTRU .SCB[PSM$L_START_PAGE]
; 2300          3214  2     OR .SCB[PSM$L_PAGE] LSSU .CLOSEST[SMBMSG$[_PAGE]
; 2301          3215  2     THEN
; 2302          3216  2         .CLOSEST
; 2303          3217  2     ELSE
; 2304          3218  2         0
; 2305          3219  2 ELSE
; 2306          3220  2     0
; 2307          3221  2
; 2308          3222  1 END;



                                          000C 00000 FIND_CHECKPOINT:
                                                            .WORD   Save R2,R3                                   ; 3166
                                       50   D4 00002        CLRL    CLOSEST                                      ; 3169
                   52          04      AC   D0 00004        MOVL    SCB, R2                                      ; 3179
                   53        017C      C2   D0 00008        MOVL    380(R2), DSB                                 ;
                   51        017C      C2   9E 0000D 1$:    MOVAB   380(R2), R1                                  ; 3184
                   51                  53   D1 00012        CMPL    DSB, R1                                      ;
                                       1F   13 00015        BEQL    4$                                           ;
                   51          0C      A3   D0 00017        MOVL    12(DSB), R1                                  ; 3187
         0224      C2          08      A1   D1 0001B        CMPL    8(R1), 548(R2)                               ; 3192
                                       0E   14 00021        BGTR    3$                                           ;
                                       50   D5 00023        TSTL    CLOSEST                                      ; 3194
                                       07   13 00025        BEQL    2$                                           ;
           08      A0          08      A1   D1 00027        CMPL    8(R1), 8(CLOSEST)                            ; 3196
                                       03   1B 0002C        BLEQU   3$                                           ;
                   50                  51   D0 0002E 2$:    MOVL    R1, CLOSEST                                  ; 3198
                   53                  63   D0 00031 3$:    MOVL    (DSB), DSB                                   ; 3202
                                       D7   11 00034        BRB     1$                                           ; 3184
                                       50   D5 00036 4$:    TSTL    CLOSEST                                      ; 3208
                                       11   13 00038        BEQL    5$                                           ;
         0224      C2        01EC      C2   D1 0003A        CMPL    492(R2), 548(R2)                             ; 3213
                                       0A   1A 00041        BGTRU   6$                                           ;
           08      A0        01EC      C2   D1 00043        CMPL    492(R2), 8(CLOSEST)                          ; 3214
                                       02   1F 00049        BLSSU   6$                                           ;
                                       50   D4 0004B 5$:    CLRL    R0                                           ; 3208
                                            04 0004D 6$:    RET                                                  ; 3222

; Routine Size:  78 bytes,    Routine Base:  CODE + 0D8B
```

```
1 %SBTTL 'GET_BUFFER - Get an output buffer (IOB)'
1 ! Functional Description:
1 !                 Allocates and initializes an IOB (Input/Output buffer
1 !                 control Block)
1 !
1 ! Formal Parameters:
1 !                 SCB      : SCB address
1 !
1 ! Implicit Inputs:
1 !                 none
1 !
1 ! Implicit Outputs:
1 !                 none
1 !
1 ! Returned Value:
1 !                 SS$_NORMAL if successful
1 !                 0 if no IOB's available
1 !
1 ! Side Effects:
1 !                 Allocates and initializes the IOB queue the first
1 !                 time this routine is called.
1 !--
1 ROUTINE GET_BUFFER (
1         SCB      : REF $BBLOCK
1         )        =
2 BEGIN
2
2 LOCAL
2         IOB      : REF $BBLOCK
2         ;
2
2 ! If their is already an IOB attached to the SCB then we are done
2 !
2 IF .SCB[PSM$A_IOB] NEQ 0
2 THEN
2     RETURN SS$_NORMAL;
2
2
2 ! If the queue has never been initialized then do it
2 !
2 IF .FLINK_ (SCB[PSM$Q_BUFFER_QUEUE]) EQL 0
2 THEN
3     BEGIN
3     INIT_QUEUE_HEADER_ (SCB[PSM$Q_BUFFER_QUEUE]);
3
3     ! Allocate as many IOB's for this SCB as specified by NUMOUTBUF
3     !
3     DECR I FROM PSM$K_NUMOUTBUF TO 1
3     DO
4         BEGIN
4         PSM$ALLOCATE_IOB (IOB, PSM$GL_MAXBUF);
4         IOB[IOB_A_CONTEXT] = .SCB;
4         INSERT_TAIL_ (IOB[IOB_Q_QLINKS], SCB[PSM$Q_BUFFER_QUEUE]);
3         END;
2     END;
2
2
```

```
; 2367          3280  2 ! Get an IOB, return if none available
; 2368          3281  2 !
; 2369          3282  2 IF REMOVE_HEAD_ (IOB, SCB[PSM$Q_BUFFER_QUEUE]) THEN RETURN 0;
; 2370          3283  2
; 2371          3284  2
; 2372          3285  2 ! Adjust the IOB address, clear the IOB flags, and attach the
; 2373          3286  2 ! IOB to the SCB.
; 2374          3287  2 !
; 2375          3288  2 IOB = .IOB - $BYTEOFFSET (IOB_Q_QLINKS);
; 2376          3289  2 IOB[IOB_L_FLAGS] = 0;
; 2377          3290  2 SCB[PSM$A_IOB] = .IOB;
; 2378          3291  2
; 2379          3292  2
; 2380          3293  2 ! Initialize the buffer descriptor
; 2381          3294  2 !
; 2382          3295  2 VECTOR [SCB[PSM$Q_OUTPUT_BUFFER], 0] = .DESC_SIZE_ (IOB[IOB_Q_BUFFER]);
; 2383          3296  2 VECTOR [SCB[PSM$Q_OUTPUT_BUFFER], 1] = .DESC_ADDR_ (IOB[IOB_Q_BUFFER]);
; 2384          3297  2
; 2385          3298  2 SS$_NORMAL
; 2386          3299  2
; 2387          3300  1 END;



                                          0004 00000 GET_BUFFER:
                                                              .WORD    Save R2                                      ; 3245
                          5E           04  C2 00002           SUBL2    #4, SP                                       ;
                          50      04   AC  D0 00005           MOVL     SCB, R0                                      ; 3256
                               01AC    C0  D5 00009           TSTL     428(R0)                                      ;
                                       64  12 0000D           BNEQ     4$                                           ;
                          50    0174   C0  9E 0000F           MOVAB    372(R0), R0                                  ; 3263
                                       60  D5 00014           TSTL     (R0)                                         ;
                                       2E  12 00016           BNEQ     2$                                           ;
                          60           50  D0 00018           MOVL     R0, (R0)                                     ; 3266
                  04      A0           50  D0 0001B           MOVL     R0, 4(R0)                                    ;
                          52           03  D0 0001F           MOVL     #3, I                                        ; 3270
                           00000000G   00  9F 00022 1$:       PUSHAB   PSM$GL_MAXBUF                                ; 3273
                                  04   AE  9F 00028           PUSHAB   IOB                                          ;
           00000000G      00           02  FB 0002B           CALLS    #2, PSM$ALLOCATE_IOB                         ;
                          51           6E  D0 00032           MOVL     IOB, R1                                      ; 3274
                  14      A1      04   AC  D0 00035           MOVL     SCB, 20(R1)                                  ;
                          50      04   AC  D0 0003A           MOVL     SCB, R0                                      ; 3275
                0178      D0           61  0E 0003E           INSQUE   (R1), @376(R0)                               ;
                          DC           52  F5 00043           SOBGTR   I, 1$                                        ; 3270
                          50      04   AC  D0 00046 2$:       MOVL     SCB, R0                                      ; 3282
                          6E    0174   D0  0F 0004A           REMQUE   @372(R0), IOB                                ;
                                       03  1C 0004F           BVC      3$                                           ;
                                       50  D4 00051           CLRL     R0                                           ;
                                           04 00053           RET                                                   ;
                          51           6E  D0 00054 3$:       MOVL     IOB, R1                                      ; 3289
                                  2C   A1  D4 00057           CLRL     44(R1)                                       ;
                          50      04   AC  D0 0005A           MOVL     SCB, R0                                      ; 3290
                01AC      C0           81  7E 0005E           MOVAQ    (R1)+, 428(R0)                               ;
                          50    01E0   C0  9E 00063           MOVAB    480(R0), R0                                  ; 3295
                          51           14  C0 00068           ADDL2    #20, R1                                      ;
```

```
                              60          61  3C 0006B          MOVZWL  (R1), (R0)                                      ;
                        04    A0    04    A1  D0 0006E          MOVL    4(R1), 4(R0)                                    ; 3296
                              50          01  D0 00073 4$:      MOVL    #1, R0                                          ; 3300
                                              04 00076          RET                                                     ;

; Routine Size:  119 bytes,    Routine Base:  CODE + 0DD9
; 2389          3301  1 %SBTTL 'HANDLER -- main signal handler'
; 2390          3302  1 ! Functional Description:
; 2391          3303  1 !               Catches signals, inhibits text expansion, and resignals
; 2392          3304  1 !
; 2393          3305  1 ! Formal Parameters:
; 2394          3306  1 !               STANDARD SIGNAL ARGUMENTS
; 2395          3307  1 !
; 2396          3308  1 ! Implicit Inputs:
; 2397          3309  1 !               none
; 2398          3310  1 !
; 2399          3311  1 ! Implicit Outputs:
; 2400          3312  1 !               none
; 2401          3313  1 !
; 2402          3314  1 ! Returned Value:
; 2403          3315  1 !               none
; 2404          3316  1 !
; 2405          3317  1 ! Side Effects:
; 2406          3318  1 !               none
; 2407          3319  1 !--
; 2408          3320  1 ROUTINE HANDLER (SIGARGS: REF BLOCK [, BYTE]) =
; 2409          3321  1
; 2410          3322  2 BEGIN
; 2411          3323  2
; 2412          3324  2 ! Disable expansion of error condition to text
; 2413          3325  2 !
; 2414          3326  2 SIGARGS [CHF$L_SIG_NAME] = .SIGARGS [CHF$L_SIG_NAME] OR STS$M_INHIB_MSG;
; 2415          3327  2
; 2416          3328  2 SS$_RESIGNAL
; 2417          3329  2
; 2418          3330  1 END;



                                         0000 00000 HANDLER:.WORD    Save nothing                                  ; 3320
                      50        04    AC  D0 00002         MOVL     SIGARGS, R0                                   ; 3326
                07    A0              10  88 00006         BISB2    #16, 7(R0)                                    ;
                      50      0918    8F  3C 0000A         MOVZWL   #2328, R0                                     ; 3330
                                          04 0000F         RET                                                    ;

; Routine Size:  16 bytes,    Routine Base:  CODE + 0E50
```

```
1 %SBTTL 'PUTMSG_ACTION - action routine for $PUTMSG call'
1 ! Functional Description:
1 !             Adds carraige control and appends the messages into
1 !             the SCB.
1 !
1 ! Formal Parameters:
1 !             Standard $PUTMSG action routine interface
1 !
1 ! Implicit Inputs:
1 !             none
1 !
1 ! Implicit Outputs:
1 !             none
1 !
1 ! Returned Value:
1 !             none
1 !
1 ! Side Effects:
1 !             The message text is appended to the appropriate descriptor
1 !             in the SCB.
1 !--
1 ROUTINE PUTMSG_ACTION (
1         MSG_DESC        : REF $BBLOCK,
1         DYN_DESC
1         ) =
2 BEGIN
2
2 BIND FORMAT = $DESCRIPTOR ('!/!AS',  %CHAR (PSM$K_CHAR_CR));
2
2 LOCAL
2         WRK_DESC:       VECTOR [2],
2         WRK_BUFF:       VECTOR [512, BYTE]
2         ;
2
2 ! Setup a work descriptor
2 !
2 WRK_DESC [0] = %ALLOCATION (WRK_BUFF);
2 WRK_DESC [1] = WRK_BUFF;
2
2
2 ! Call FAO to add carriage control
2 !
2 $FAO (FORMAT, WRK_DESC, WRK_DESC, .MSG_DESC);
2
2
2 ! Append the resulting message to the specified descriptor
2 !
2 SIGNAL_IF_ERROR_ (STR$APPEND (.DYN_DESC, WRK_DESC));
2
2 RETURN 0;
2
1 END;


                    53  41  21  2F  21  00E60 P.AAS:   .ASCII  \!/!AS\
                                    0D  00E65          .ASCII  <13>
```

```
                                                00E66          .BLKB   2                                             ;
                                    00000006    00E68 P.AAR:   .LONG   6                                             ;
                                    00000000'   00E6C          .ADDRESS P.AAS                                        ;

                                                      FORMAT=          P.AAR
                                                               .EXTRN  SYS$FAO

                                          0004 00000 PUTMSG_ACTION:
                                                               .WORD   Save R2                                       ; 3352
                        5E      FDF8   CE   9E 00002           MOVAB   -520(SP), SP                                  ;
                FB      AD      0200   8F   3C 00007           MOVZWL  #512, WRK_DESC                                ; 3367
                FC      AD             6E   9E 0000D           MOVAB   WRK_BUFF, WRK_DESC+4                          ; 3368
                                04     AC   DD 00011           PUSHL   MSG_DESC                                      ; 3373
                                FB     AD   9F 00014           PUSHAB  WRK_DESC                                      ;
                                FB     AD   9F 00017           PUSHAB  WRK_DESC                                      ;
                                DB     AF   9F 0001A           PUSHAB  FORMAT                                        ;
        00000000G       00             04   FB 0001D           CALLS   #4, SYS$FAO                                   ;
                                FB     AD   9F 00024           PUSHAB  WRK_DESC                                      ; 3378
                                08     AC   DD 00027           PUSHL   DYN_DESC                                      ;
        00000000G       00             02   FB 0002A           CALLS   #2, STR$APPEND                                ;
                        52             50   D0 00031           MOVL    R0, STATUS                                    ;
                        09             52   E8 00034           BLBS    STATUS, 1$                                    ;
                                       52   DD 00037           PUSHL   STATUS                                        ;
        00000000G       00             01   FB 00039           CALLS   #1, LIB$SIGNAL                                ;
                                       50   D4 00040 1$:       CLRL    R0                                            ; 3380
                                            04 00042           RET                                                   ; 3382

; Routine Size:  67 bytes,    Routine Base:  CODE + 0E70
```

```
1 %SBTTL 'RESUME_SERVICE - Resume a previously suspended service'
1 ! Functional Description:
1 !             Resumes the input service at the top of the service
1 !             stack and resets the SCB values that were in effect
1 !             when the service was suspended.
1 !
1 ! Formal Parameters:
1 !             SCB     : SCB ADDRESS
1 !
1 ! Implicit Inputs:
1 !             Input service queue header
1 !
1 ! Implicit Outputs:
1 !             Context values that are preserved when a service is
1 !             suspended are restored.
1 !
1 ! Returned Value:
1 !             none
1 !
1 ! Side Effects:
1 !             The service is popped from the input service stack.
1 !--
1 ROUTINE RESUME_SERVICE (
1         SCB     : REF $BBLOCK
1         )       : NOVALUE =
2 BEGIN
2
2 LOCAL
2         DSB : REF $BBLOCK
2         ;
2
2 ! Decrement the depth and check for coding error
2 !
2 DECREMENT_ (SCB[PSM$B_INPUT_DEPTH]);
2 IF .SCB[PSM$B_INPUT_DEPTH] LSS 0
2 THEN
2     CODEERR_ ;
2
2
2 ! Release any dynamic memory of current stream
2 !
2 CLEAR_STRING_ (SCB[PSM$Q_INPUT_RECORD]);
2 CLEAR_STRING_ (SCB[PSM$Q_USER_RECORD]);
2
2
2 ! Get the context block for the previous stream
2 !
2 IF REMOVE_HEAD_ (DSB, SCB[PSM$Q_INPUT_QUEUE]) THEN CODEERR_ ;
2 DSB = .DSB - $BYTEOFFSET (DSB_Q_QLINKS);
2
2
2 ! Overlay the context area in the SCB
2 !
2 CH$MOVE (PSM$S_SERVICE_CONTEXT, .DESC_ADDR_ (DSB[DSB_Q_DESC]),
2     SCB[PSM$R_SERVICE_CONTEXT]);
2
2
```

```
; 2530          3440  2 ! Release the context block
; 2531          3441  2 !
; 2532          3442  2 PSM$DEALLOCATE_DSB (.DSB);
; 2533          3443  2
; 2534          3444  1 END;



                                         01FC 00000 RESUME_SERVICE:
                                                             .WORD    Save R2,R3,R4,R5,R6,R7,R8                     ; 3405
                          58 00000000G  00  9E 00002         MOVAB    LIB$STOP, R8                                  ;
                          57 00000000G  00  9E 00009         MOVAB    STR$FREE1_DX, R7                              ;
                          52        04  AC  D0 00010         MOVL     SCB, R2                                       ; 3416
                          50      02A5  C2  9E 00014         MOVAB    677(R2), R0                                   ;
                                        60  97 00019         DECB     (R0)                                          ;
                                        0B  18 0001B         BGEQ     1$                                            ; 3417
                                        01  DD 0001D         PUSHL    #1                                            ; 3418
                             01061154   8F  DD 0001F         PUSHL    #17174868                                     ;
                          68            02  FB 00025         CALLS    #2, LIB$STOP                                  ;
                          01      0263  C2  91 00028 1$:     CMPB     611(R2), #1                                   ; 3424
                                        11  1A 0002D         BGTRU    2$                                            ;
                          50      0260  C2  9E 0002F         MOVAB    608(R2), R0                                   ;
                          60 020E0000   8F  D0 00034         MOVL     #34471936, (R0)                               ;
                                    04  A0  D4 0003B         CLRL     4(R0)                                         ;
                                        10  11 0003E         BRB      3$                                            ;
                          50            52  D0 00040 2$:     MOVL     R2, R0                                        ;
                                  0260  C0  B5 00043         TSTW     608(R0)                                       ;
                                        07  13 00047         BEQL     3$                                            ;
                                  0260  C2  9F 00049         PUSHAB   608(R2)                                       ;
                          67            01  FB 0004D         CALLS    #1, STR$FREE1_DX                              ;
                          01      0273  C2  91 00050 3$:     CMPB     627(R2), #1                                   ; 3425
                                        11  1A 00055         BGTRU    4$                                            ;
                          50      0270  C2  9E 00057         MOVAB    624(R2), R0                                   ;
                          60 020E0000   8F  D0 0005C         MOVL     #34471936, (R0)                               ;
                                    04  A0  D4 00063         CLRL     4(R0)                                         ;
                                        10  11 00066         BRB      5$                                            ;
                          50            52  D0 00068 4$:     MOVL     R2, R0                                        ;
                                  0270  C0  B5 0006B         TSTW     624(R0)                                       ;
                                        07  13 0006F         BEQL     5$                                            ;
                                  0270  C2  9F 00071         PUSHAB   624(R2)                                       ;
                          67            01  FB 00075         CALLS    #1, STR$FREE1_DX                              ;
                          56      0184  D2  0F 00078 5$:     REMQUE   @388(R2), DSB                                 ; 3430
                                        0B  1C 0007D         BVC      6$                                            ;
                                        01  DD 0007F         PUSHL    #1                                            ;
                             01061154   8F  DD 00081         PUSHL    #17174868                                     ;
                          68            02  FB 00087         CALLS    #2, LIB$STOP                                  ;
                          50        04  AC  D0 0008A 6$:     MOVL     SCB, R0                                       ; 3437
          0260    C0        0C  B6      1E  28 0008E         MOVC3    #30, @12(DSB), 608(R0)                        ;
                                        56  DD 00095         PUSHL    DSB                                           ; 3442
                    00000000G   00      01  FB 00097         CALLS    #1, PSM$DEALLOCATE_DSB                        ;
                                            04 0009E         RET                                                    ; 3444

; Routine Size:  159 bytes,    Routine Base:  CODE + 0EB3
```

```
1 %SBTTL 'SAVE_CHECKPOINT - Build a checkpoint item'
1 ! Functional Description:
1 !                 Builds a checkpoint item from values in the SCB and from
1 !                 a READ_KEY operation to the current input service.
1 !
1 ! Formal Parameters:
1 !                 SCB       : SCB address
1 !
1 ! Implicit Inputs:
1 !                 none
1 !
1 ! Implicit Outputs:
1 !                 none
1 !
1 ! Returned Value:
1 !                 none
1 !
1 ! Side Effects:
1 !                 none
1 !--
1 ROUTINE SAVE_CHECKPOINT (
1         SCB       : REF $BBLOCK
1         )         : NOVALUE =
2 BEGIN
2
2 LOCAL
2         CKP_DESC          : VECTOR [2],
2         KEY_DESC          : VECTOR [2] PRESET ([0]=0, [1]=0)
2         ;
2
2 BIND
2         IOB = .SCB[PSM$A_IOB] : $BBLOCK,                     ! Current output blcok
2         CKP = IOB[IOB_T_CHECKPOINT_DATA] : $BBLOCK           ! Checkpoint area in IOB
2         ;
2
3 BEGIN
3
3 ! Locate the current input service
3 !
3 BIND SERVICE = PSM$SRV[.SCB[PSM$B_SERVICE_INDEX],0,0,0,0] : $BBLOCK;
3 LOCAL FUNCTION_STATUS;
3
3
3 ! Call the current input service to obtain the record key
3 !
3 FUNCTION_STATUS = BLISS (
3     .SERVICE[SRV_A_SERVICE],              !  - current input service
3     SCB,                                  !  - SCB address by reference
3     SCB[PSM$R_USER_CONTEXT_AREA],         !  - user context area
3     UPLIT (PSM$K_GET_KEY),                !  - GET_KEY function
3     KEY_DESC,                             !  - output key desc
3     0);                                   !  - <not used>
3
3 ! Case on the status
3 !
3 SELECTONEU .FUNCTION_STATUS OF
3     SET
```

```

            ! Asynchronous read_key operations not allowed
            !

            [PSM$_PENDING]:
                CODEERR_ ;


            ! If not supported, then return that as our status
            !

            [PSM$_FUNNOTSUP]:
                RETURN PSM$_FUNNOTSUP;


            ! If errors then store them and return the error
            !

            [OTHERWISE]:
                IF NOT .FUNCTION_STATUS
                THEN
                    BEGIN
                    PSM$STORE_ERRORS (.SCB, .FUNCTION_STATUS);
                    RETURN .FUNCTION_STATUS;
                    END;
            TES;
        END;

        ! We have a key -- check the size and copy it into
        !
        IF .KEY_DESC[0] GTRU SMBMSG$S_USER_KEY THEN CODEERR_ ;
        CH$COPY (.KEY_DESC[0], .KEY_DESC[1], 0,
            SMBMSG$S_USER_KEY, CKP[SMBMSG$Q_USER_KEY]);


        ! Build the rest of the checkpoint
        !
        CKP[SMBMSG$B_CHECKPOINT_LEVEL] = SMBMSG$K_STRUCTURE_LEVEL;
        CKP[SMBMSG$W_OFFSET] = .SCB_SIZE_ (USER_RECORD) - .SCB_SIZE_ (INPUT_RECORD);
        CKP[SMBMSG$L_CARCON] = .SCB[PSM$L_CARCON];
        CKP[SMBMSG$L_PAGE] = .SCB[PSM$L_PAGE];
        CKP[SMBMSG$L_RECORD_NUMBER] = .SCB[PSM$L_RECORD_NUMBER];


        ! Mark this IOB as having a checkpoint assoicated with it.
        !
        IOB[IOB_V_CHECKPOINT_PENDING] = 1;


        ! Build a descriptor of the checkpoint
        !
        CKP_DESC[0] = SMBMSG$S_CHECKPOINT_DATA;
        CKP_DESC[1] = CKP;


        ! Place it in the checkpoint queue
```

```
; 2650        3559  2 !
; 2651        3560  2 ENQUEUE_CHECKPOINT (.SCB, CKP_DESC[0]);
; 2652        3561  2
; 2653        3562  2
; 2654        3563  2 SS$_NORMAL
; 2655        3564  2
; 2656        3565  1 END;


                                              00F52          .BLKB   2
                                    00000006  00F54 P.AAT:   .LONG   6                                          ;


                                         03FC 00000 SAVE_CHECKPOINT:
                                                             .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9             ; 3465
                        59        F7    AF  9E 00002         MOVAB   P.AAT, R9                                ;
                        58 00000000G    00  9E 00006         MOVAB   LIB$STOP, R8                             ;
                        5E              0C  C2 0000D         SUBL2   #12, SP                                  ;
                                        7E  D4 00010         CLRL    KEY_DESC                                 ; 3472
                                  04    AE  D4 00012         CLRL    KEY_DESC+4                               ;
                        50        04    AC  D0 00015         MOVL    SCB, R0                                  ; 3476
                        57      01AC    C0  D0 00019         MOVL    428(R0), R7                              ;
                        56        30    A7  9E 0001E         MOVAB   48(R7), R6                               ; 3477
                        51      027D    C0  9A 00022         MOVZBL  637(R0), R1                              ; 3484
                        51              10  C4 00027         MULL2   #16, R1                                  ;
                           00000000G00041   9F 0002A         PUSHAB  PSM$SRV[R1]                              ; 3491
                        51              9E  D0 00031         MOVL    @(SP)+, R1                               ;
                                        7E  D4 00034         CLRL    -(SP)                                    ; 3493
                                  04    AE  9F 00036         PUSHAB  KEY_DESC                                 ;
                                        59  DD 00039         PUSHL   R9                                       ; 3494
                                02D0    C0  9F 0003B         PUSHAB  720(R0)                                  ; 3493
                                  04    AC  9F 0003F         PUSHAB  SCB                                      ; 3490
                        61              05  FB 00042         CALLS   #5, (R1)                                 ; 3493
                        52              50  D0 00045         MOVL    R0, FUNCTION_STATUS                      ;
              00000000G 8F              52  D1 00048         CMPL    FUNCTION_STATUS, #PSM$_PENDING           ; 3507
                                        0D  12 0004F         BNEQ    1$                                       ;
                                        01  DD 00051         PUSHL   #1                                       ;
                           01061154     8F  DD 00053         PUSHL   #17174868                                ;
                        68              02  FB 00059         CALLS   #2, LIB$STOP                             ;
                                        17  11 0005C         BRB     2$                                       ;
              00000000G 8F              52  D1 0005E 1$:     CMPL    FUNCTION_STATUS, #PSM$_FUNNOTSUP         ; 3514
                                        5F  13 00065         BEQL    4$                                       ;
                        0B              52  E8 00067         BLBS    FUNCTION_STATUS, 2$                      ; 3522
                                        52  DD 0006A         PUSHL   FUNCTION_STATUS                          ; 3525
                                  04    AC  DD 0006C         PUSHL   SCB                                      ;
                   FB1D CF              02  FB 0006F         CALLS   #2, PSM$STORE_ERRORS                     ;
                                            04 00074         RET                                              ; 3526
                        08              6E  D1 00075 2$:     CMPL    KEY_DESC, #8                             ; 3533
                                        0B  1B 00078         BLEQU   3$                                       ;
                                        01  DD 0007A         PUSHL   #1                                       ;
                           01061154     8F  DD 0007C         PUSHL   #17174868                                ;
                        68              02  FB 00082         CALLS   #2, LIB$STOP                             ;
  08            00      04 BE           6E  2C 00085 3$:     MOVC5   KEY_DESC, @KEY_DESC+4, #0, #8, 16(R6)    ; 3535
                                  10    A6     0008B                                                          ;
                     01 A6              01  90 0008D         MOVB    #1, 1(R6)                                ; 3540
```

```
                                 50          04    AC   D0 00091           MOVL     SCB, R0                                    ; 3541
                 02    A6      0270  C0    0260    C0   A3 00095           SUBW3    608(R0), 624(R0), 2(R6)                    ;
                                 04  A6    0278    C0   D0 0009E           MOVL     632(R0), 4(R6)                             ; 3542
                                 08  A6    01EC    C0   D0 000A4           MOVL     492(R0), 8(R6)                             ; 3543
                                 0C  A6    026C    C0   D0 000AA           MOVL     620(R0), 12(R6)                            ; 3544
                                 2C  A7            01   88 000B0           BISB2    #1, 44(R7)                                 ; 3549
                                 08  AE            18   D0 000B4           MOVL     #24, CKP_DESC                              ; 3554
                                 0C  AE            56   D0 000B8           MOVL     R6, CKP_DESC+4                             ; 3555
                                             08    AE   9F 000BC           PUSHAB   CKP_DESC                                   ; 3560
                                                   50   DD 000BF           PUSHL    R0                                         ;
                               FC7D  CF            02   FB 000C1           CALLS    #2, ENQUEUE_CHECKPOINT                     ;
                                                        04 000C6 4$:       RET                                                 ; 3565

; Routine Size:  199 bytes,    Routine Base:  CODE + 0F58
```

```
1 %SBTTL 'SCHEDULE_SERVICE -- determine the next input service to process'
1 ! Functional Description:
1 !             Looks for an input service to process.  The primary list
1 !             of services is established by a bit vector.  Additional
1 !             sources of input are page headers, page setup, included modules,
1 !             and previously suspended input services.
1 !
1 ! Formal Parameters:
1 !             SCB     : SCB address
1 !
1 ! Implicit Inputs:
1 !             none
1 !
1 ! Implicit Outputs:
1 !             none
1 !
1 ! Returned Value:
1 !             SS$_NORMAL      - Service located
1 !             PSM$_EOF        - No input services remain
1 !
1 ! Side Effects:
1 !             An input service may be dequeued from the input stack,
1 !             or removed from the outstanding service list.
1 !--
1
1 ROUTINE SCHEDULE_SERVICE (
1       SCB     : REF $BBLOCK
1       )       =
2 BEGIN
2
2 BIND
2       LIST    = SCB[PSM$L_SERVICE_LIST] : BITVECTOR
2       ;
2
2 LOCAL
2       PIDX    : INITIAL (0)           ! Index into service list
2       ;
2
2 ! Reset values for new input service
2 !
2 SCB[PSM$L_RECORD_NUMBER] = 0;
2 SCB[PSM$V_READ_OFFSET] = 0;
2 SCB[PSM$V_FIRST_RECORD] = 1;
2 SCB[PSM$B_SERVICE_INDEX] = 0;
2
2
2 ! If there are any pending modules then select the LIBRARY_INPUT service
2 ! to process them.
2 !
2 IF STRIP_COMMA_DELIMITED_ITEM (SCB[PSM$Q_MODULE_LIST], SCB[PSM$Q_MODULE_NAME])
2 THEN
3     BEGIN
3     SCB[PSM$B_SERVICE_INDEX] = PSM$K_LIBRARY_INPUT;
3     RETURN SS$_NORMAL;
2     END;
2
```

```
2 ! If page setup has been requested then schedule it
2 !
2 IF TESTBITSC (LIST[PSM$K_PAGE_SETUP])
2 THEN
3     BEGIN
3     SCB[PSM$B_SERVICE_INDEX] = PSM$K_PAGE_SETUP;
3     RETURN SS$_NORMAL;
2     END;
2
2
2 ! Similarily, if page header has been requested then schedule it
2 !
2 IF TESTBITSC (LIST[PSM$K_PAGE_HEADER])
2 THEN
3     BEGIN
3     SCB[PSM$B_SERVICE_INDEX] = PSM$K_PAGE_HEADER;
3     RETURN SS$_NORMAL;
2     END;
2
2
2 ! If there is a suspended input service then resume it
2 !
2 IF .SCB[PSM$B_INPUT_DEPTH] GTRU 0
2 THEN
3     BEGIN
3     RESUME_SERVICE (.SCB);
3     RETURN SS$_NORMAL;
2     END;
2
2
2 ! This is a brand new input service -- reset values
2 !
2 SCB[PSM$L_PAGE] = 1;
2 SCB[PSM$L_PRINT_FLAGS] = 0;
2 SCB[PSM$L_L_MARGIN] = 0;
2 SCB[PSM$L_T_MARGIN] = 0;
2
2
2 ! Scan the service list for a pending input service
2 !
2 UNTIL FFS (PIDX, UPLIT (PSM$K_MAX), LIST, PIDX) ! False until list empty
2 DO
3     BEGIN
3     SCB[PSM$B_SERVICE_INDEX] = .PIDX;
3     LIST[.PIDX] = 0;
3     IF .PSM$SRV[.PIDX, SRV_A_SERVICE] NEQ 0
3     THEN
3         RETURN SS$_NORMAL;
2     END;
2
2
2 ! No service found, return EOF
2 !
2 PSM$_EOF
2
1 END;
```

```
                                                   0101F          .BLKB    1
                                          00000017 01020 P.AAU:   .LONG    23                                              ;


                                              003C 00000 SCHEDULE_SERVICE:
                                                                  .WORD    Save R2,R3,R4,R5                                ; 3591
                            53       04     AC  D0 00002          MOVL     SCB, R3                                         ; 3597
                            55     0218     C3  9E 00006          MOVAB    536(R3), R5                                     ;
                                            52  D4 0000B          CLRL     PIDX                                            ;
                                   026C     C3  D4 0000D          CLRL     620(R3)                                         ; 3606
                       11   A3              02  8A 00011          BICB2    #2, 17(R3)                                      ; 3607
                       10   A3              20  88 00015          BISB2    #32, 16(R3)                                     ; 3608
                            54     027D     C3  9E 00019          MOVAB    637(R3), R4                                     ; 3609
                                            64  94 0001E          CLRB     (R4)                                            ;
                                   01D4     C3  9F 00020          PUSHAB   468(R3)                                         ; 3615
                                   01CC     C3  9F 00024          PUSHAB   460(R3)                                         ;
                    0000V   CF              02  FB 00028          CALLS    #2, STRIP_COMMA_DELIMITED_ITEM                  ;
                            05              50  E9 0002D          BLBC     R0, 1$                                          ;
                            64              03  90 00030          MOVB     #3, (R4)                                        ; 3618
                                            50  11 00033          BRB      7$                                              ; 3619
               05           65              01  E5 00035 1$:      BBCC     #1, (R5), 2$                                    ; 3625
                            64              01  90 00039          MOVB     #1, (R4)                                        ; 3628
                                            47  11 0003C          BRB      7$                                              ; 3629
               05           65              02  E5 0003E 2$:      BBCC     #2, (R5), 3$                                    ; 3635
                            64              02  90 00042          MOVB     #2, (R4)                                        ; 3638
                                            3E  11 00045          BRB      7$                                              ; 3639
                                   02A5     C3  95 00047 3$:      TSTB     677(R3)                                         ; 3645
                                            09  13 0004B          BEQL     4$                                              ;
                                            53  DD 0004D          PUSHL    R3                                              ; 3648
                    FE3B    CF              01  FB 0004F          CALLS    #1, RESUME_SERVICE                              ;
                                            2F  11 00054          BRB      7$                                              ; 3649
                    01EC    C3              01  D0 00056 4$:      MOVL     #1, 492(R3)                                     ; 3655
                                   0204     C3  D4 0005B          CLRL     516(R3)                                         ; 3656
                                   01BC     C3  D4 0005F          CLRL     444(R3)                                         ; 3657
                                   0230     C3  D4 00063          CLRL     560(R3)                                         ; 3658
      52       65       91  AF              52  EA 00067 5$:      FFS      PIDX, P.AAU, (R5), PIDX                         ; 3663
                                            1A  13 0006D          BEQL     8$                                              ;
                            64              52  90 0006F          MOVB     PIDX, (R4)                                      ; 3666
               00           65              52  E5 00072          BBCC     PIDX, (R5), 6$                                  ; 3667
               50           52              04  78 00076 6$:      ASHL     #4, PIDX, R0                                    ; 3668
                                 00000000G00040 9F 0007A          PUSHAB   PSM$SRV[R0]                                     ;
                                            9E  D5 00081          TSTL     @(SP)+                                          ;
                                            E2  13 00083          BEQL     5$                                              ;
                            50              01  D0 00085 7$:      MOVL     #1, R0                                          ; 3670
                                                04 00088          RET                                                      ;
                            50 00000000G    8F  D0 00089 8$:      MOVL     #PSM$_EOF, R0                                   ; 3678
                                                04 00090          RET                                                      ;

; Routine Size:  145 bytes,    Routine Base:  CODE + 1024
```

```
1 %SBTTL 'SEARCH_FOR_STRING - Search for a string in a buffer'
1 ! Functional Description:
1 !                 This routine looks for a search string in the current
1 !                 input record.  It maintains context across calls so that
1 !                 strings that cross record boundaries can be located.
1 !
1 ! Formal Parameters:
1 !                 SCB     : SCB address
1 !                 KEY     : descriptor of search key
1 !                 TARGET  : descriptor of input record
1 !
1 ! Implicit Inputs:
1 !                 SCB[PSM$Q_SEARCH_CONTEXT]       - context from last call
1 !
1 ! Implicit Outputs:
1 !                 none
1 !
1 ! Returned Value:
1 !                 SS$_NORMAL      - the KEY was found in the TARGET
1 !                 0               - KEY was not found
1 ! Side Effects:
1 !                 none
1 !--
1 GLOBAL ROUTINE SEARCH_FOR_STRING (
1         SCB     : REF $BBLOCK,
1         KEY     : REF $BBLOCK,
1         TARGET  : REF $BBLOCK
1         ) =
2 BEGIN
2
2 LOCAL   PTR;
2
2 ! Append the input record to the context from the last call
2 !
2 STR$APPEND (SCB[PSM$Q_SEARCH_CONTEXT], .TARGET);
2
2
2 ! Compress white space (blanks and tabs) to a single space and upcase
2 !
2 BAS$EDIT (SCB[PSM$Q_SEARCH_CONTEXT], SCB[PSM$Q_SEARCH_CONTEXT], EDIT_MASK);
2
2
2 ! Look for the key as a substring of the target
2 !
2 PTR = CH$FIND_SUB (
2         .SCB_SIZE_ (SEARCH_CONTEXT),    ! Target appended to remainder
2         .SCB_ADDR_ (SEARCH_CONTEXT),
2         .DESC_SIZE_ (.KEY),             ! Search key
2         .DESC_ADDR_ (.KEY)
2                 );
2
2
2 ! Extract the last few characters of the input record as the context
2 ! for the next call
2 !
2 STR$RIGHT (
2         SCB[PSM$Q_SEARCH_CONTEXT],
2         SCB[PSM$Q_SEARCH_CONTEXT],
; 2829          3736  2             %REF (.SCB_SIZE_ (SEARCH_CONTEXT) - .DESC_SIZE_ (.KEY) + 1)
; 2830          3737  2             );
; 2831          3738  2
; 2832          3739  2
; 2833          3740  2   ! Return 0 if not found, SS$_normal if located
; 2834          3741  2   !
; 2835          3742  2   IF CH$FAIL (.PTR)
; 2836          3743  2   THEN
; 2837          3744  2       0
; 2838          3745  2   ELSE
; 2839          3746  2       SS$_NORMAL
; 2840          3747  2
; 2841          3748  1   END;



                                                       007C 00000         .ENTRY  SEARCH_FOR_STRING, Save R2,R3,R4,R5,R6      ; 3702
                                5E                  04   C2 00002         SUBL2   #4, SP                                     ;
                                          0C        AC   DD 00005         PUSHL   TARGET                                     ; 3713
                                50        04        AC   D0 00008         MOVL    SCB, R0                                    ;
                                55      0210        C0   9E 0000C         MOVAB   528(R0), R5                                ;
                                                    55   DD 00011         PUSHL   R5                                         ;
                      00000000G 00                  02   FB 00013         CALLS   #2, STR$APPEND                             ;
                                                    30   DD 0001A         PUSHL   #48                                        ; 3718
                                                    55   DD 0001C         PUSHL   R5                                         ;
                                                    55   DD 0001E         PUSHL   R5                                         ;
                      00000000G 00                  03   FB 00020         CALLS   #3, BAS$EDIT                               ;
                                54        08        AC   D0 00027         MOVL    KEY, R4                                    ; 3726
                                56                  64   3C 0002B         MOVZWL  (R4), R6                                   ;
     04   B5            65        04  B4            56   39 0002E         MATCHC  R6, a4(R4), (R5), a4(R5)                   ; 3727
                                                    03   13 00035         BEQL    1$                                         ;
                                53                  56   D0 00037         MOVL    R6, R3                                     ;
                                53                  56   C2 0003A  1$:    SUBL2   R6, R3                                     ;
                                50                  65   3C 0003D         MOVZWL  (R5), R0                                   ; 3736
                                51                  64   3C 00040         MOVZWL  (R4), R1                                   ;
                                50                  51   C2 00043         SUBL2   R1, R0                                     ;
                                6E        01        A0   9E 00046         MOVAB   1(R0), (SP)                                ;
                                        4020        8F   BB 0004A         PUSHR   #^M<R5,SP>                                 ; 3735
                                                    55   DD 0004E         PUSHL   R5                                         ;
                      00000000G 00                  03   FB 00050         CALLS   #3, STR$RIGHT                              ;
                                                    53   D5 00057         TSTL    PTR                                        ; 3742
                                                    03   12 00059         BNEQ    2$                                         ;
                                                    50   D4 0005B         CLRL    R0                                         ;
                                                         04 0005D         RET                                                ;
                                50                  01   D0 0005E  2$:    MOVL    #1, R0                                     ;
                                                         04 00061         RET                                                ; 3748

; Routine Size:  98 bytes,    Routine Base:  CODE + 10B5
```

```
1 %SBTTL 'STRIP_COMMA_DELIMITED_ITEM -- remove item from comma separate list'
1 ! Functional Description:
1 !             This routine removes one item from the front of a comma
1 !             separated list.
1 !
1 ! Formal Parameters:
1 !             INPUT   : descriptor of input list
1 !             OUTPUT  : removed item
1 !
1 ! Implicit Inputs:
1 !             none
1 !
1 ! Implicit Outputs:
1 !             The INPUT list is rewritten with the item removed
1 !
1 ! Returned Value:
1 !             none
1 !
1 ! Side Effects:
1 !             none
1 !--
1 ROUTINE STRIP_COMMA_DELIMITED_ITEM (
1       INPUT   : REF $BBLOCK,
1       OUTPUT  : REF $BBLOCK
1       ) =
2 BEGIN
2
2 LOCAL PTR;
2
2 ! If nothing to do then return
2 !
2 IF .DESC_SIZE_ (.INPUT) EQL 0 THEN RETURN 0;
2
2
2 ! Locate the first comma or end of string
2 !
2 PTR = CH$FIND_CH (.DESC_SIZE_ (.INPUT), .DESC_ADDR_ (.INPUT), %C ',');
2
2
2 ! If no comma found the the entire input string is the resultant item
2 ! and the input descriptor can be released
2 !
2 IF CH$FAIL (.PTR)
2 THEN
3     BEGIN
3     COPY_DX_DX_ (.INPUT, .OUTPUT);
3     STR$FREE1_DX (.INPUT);
3     END
2 ELSE
2     ! Comma found -- move the item from input list to output list
2     !
3     BEGIN
3     PTR = .PTR - .DESC_ADDR_ (.INPUT);
3     STR$LEFT (.OUTPUT, .INPUT, PTR);
3     PTR = .PTR + 2;
3     STR$RIGHT (.INPUT, .INPUT, PTR);
2     END;
```

```
; 2900          3806  2
; 2901          3807  2
; 2902          3808  2  ! Return success
; 2903          3809  2  !
; 2904          3810  2  SS$_NORMAL
; 2905          3811  2
; 2906          3812  1  END;



                                               000C 00000 STRIP_COMMA_DELIMITED_ITEM:
                                                                  .WORD   Save R2,R3                              ; 3770
                           5E           04  C2 00002              SUBL2   #4, SP
                           52      04   AC  D0 00005              MOVL    INPUT, R2                               ; 3780
                                        62  B5 00009              TSTW    (R2)
                                        03  12 0000B              BNEQ    1$
                                        50  D4 0000D              CLRL    R0
                                            04 0000F              RET
           04   B2         62           2C  3A 00010 1$:          LOCC    #44, (R2), @4(R2)                       ; 3785
                                        02  12 00015              BNEQ    2$
                                        51  D4 00017              CLRL    R1
                           6E           51  D0 00019 2$:          MOVL    R1, PTR
                                        26  12 0001C              BNEQ    4$                                      ; 3791
                                        52  DD 0001E              PUSHL   R2                                      ; 3794
                                   08   AC  DD 00020              PUSHL   OUTPUT
                 00000000G 00           02  FB 00023              CALLS   #2, STR$COPY_DX
                           53           50  D0 0002A              MOVL    R0, STATUS
                           09           53  E8 0002D              BLBS    STATUS, 3$
                                        53  DD 00030              PUSHL   STATUS
                 00000000G 00           01  FB 00032              CALLS   #1, LIB$SIGNAL
                                        52  DD 00039 3$:          PUSHL   R2                                      ; 3795
                 00000000G 00           01  FB 0003B              CALLS   #1, STR$FREE1_DX
                                        22  11 00042              BRB     5$                                      ; 3791
                           6E      04   A2  C2 00044 4$:          SUBL2   4(R2), PTR                              ; 3801
                                 4004   8F  BB 00048              PUSHR   #^M<R2,SP>                              ; 3802
                                   08   AC  DD 0004C              PUSHL   OUTPUT
                 00000000G 00           03  FB 0004F              CALLS   #3, STR$LEFT
                           6E           02  C0 00056              ADDL2   #2, PTR                                 ; 3803
                                 4004   8F  BB 00059              PUSHR   #^M<R2,SP>                              ; 3804
                                        52  DD 0005D              PUSHL   R2
                 00000000G 00           03  FB 0005F              CALLS   #3, STR$RIGHT
                           50           01  D0 00066 5$:          MOVL    #1, R0                                  ; 3812
                                            04 00069              RET

; Routine Size:  106 bytes,    Routine Base:  CODE + 1117
```

```
  1 %SBTTL 'SUSPEND_SERVICE -- suspend the current input service'
  1 ! Functional Description:
  1 !               Suspends the current input service by placing its
  1 !               context on an input service stack.
  1 !
  1 ! Formal Parameters:
  1 !               SCB     : SCB address
  1 !
  1 ! Implicit Inputs:
  1 !               none
  1 !
  1 ! Implicit Outputs:
  1 !               none
  1 !
  1 ! Returned Value:
  1 !               none
  1 !
  1 ! Side Effects:
  1 !               The current service is placed on the stack
  1 !--
  1 GLOBAL ROUTINE SUSPEND_SERVICE (
  1         SCB     : REF $BBLOCK
  1         ) : NOVALUE =
  2 BEGIN
  2
  2 LOCAL
  2         DSB : REF $BBLOCK
  2         ;
  2
  2
  2 ! Increment the stack depth and check for overflow
  2 !
  2 INCREMENT_ (SCB[PSM$B_INPUT_DEPTH]);
  2 IF .SCB[PSM$B_INPUT_DEPTH] GTR 15
  2 THEN
  3     BEGIN
  3     PSM$STORE_ERRORS (.SCB, PSM$_TOOMANYLEV, 1, .SCB[PSM$L_RECORD_NUMBER]);
  3     RETURN;
  3     END;
  2
  2
  2 ! Get a Dynamic String control Block and copy the service context area into it.
  2 !
  2 PSM$ALLOCATE_DSB (DSB);
P 2 COPY_R_DX_ (UPLIT WORD (PSM$S_SERVICE_CONTEXT), SCB[PSM$R_SERVICE_CONTEXT],
  2     DSB[DSB_Q_DESC]);
  2
  2
  2 ! Place it in the input queue
  2 !
  2 INSERT_HEAD_ (DSB[DSB_Q_QLINKS], SCB[PSM$Q_INPUT_QUEUE]);
  2
  2
  2 ! Clear the service context area
  2 !
  2 CH$FILL (0, PSM$S_SERVICE_CONTEXT, SCB[PSM$R_SERVICE_CONTEXT]);
  2
```

```
; 2965          3870  1 END;



                                              01181         .BLKB    1
                                        001E  01182 P.AAV:  .WORD    30                                                ;



                                         003C 00000         .ENTRY   SUSPEND_SERVICE, Save R2,R3,R4,R5               ; 3833
                          5E          04  C2  00002         SUBL2    #4, SP                                           ;
                          52      04  AC  D0  00005         MOVL     SCB, R2                                          ; 3845
                          50    02A5  C2  9E  00009         MOVAB    677(R2), R0                                      ;
                                      60  96  0000E         INCB     (R0)                                             ;
                          0F          60  91  00010         CMPB     (R0), #15                                        ; 3846
                                      14  15  00013         BLEQ     1$                                               ;
                                026C  C2  DD  00015         PUSHL    620(R2)                                          ; 3849
                                      01  DD  00019         PUSHL    #1                                               ;
                           00000000G  8F  DD  0001B         PUSHL    #PSM$_TOOMANYLEV                                 ;
                                      52  DD  00021         PUSHL    R2                                               ;
                  F93D  CF            04  FB  00023         CALLS    #4, PSM$STORE_ERRORS                             ;
                                          04  00028         RET                                                       ; 3848
                                      5E  DD  00029 1$:     PUSHL    SP                                               ; 3856
             00000000G  00            01  FB  0002B         CALLS    #1, PSM$ALLOCATE_DSB                             ;
                                0260  C2  9F  00032         PUSHAB   608(R2)                                          ; 3858
                                  C5  AF  9F  00036         PUSHAB   P.AAV                                            ;
                 7E      08  AE           08  C1  00039     ADDL3    #8, DSB, -(SP)                                   ;
             00000000G  00            03  FB  0003E         CALLS    #3, STR$COPY_R                                   ;
                        53            50  D0  00045         MOVL     R0, STATUS                                       ;
                        09            53  E8  00048         BLBS     STATUS, 2$                                       ;
                                      53  DD  0004B         PUSHL    STATUS                                           ;
             00000000G  00            01  FB  0004D         CALLS    #1, LIB$SIGNAL                                   ;
                  0184  D2        00  BE  0E  00054 2$:     INSQUE   @DSB, @388(R2)                                   ; 3863
                        50        04  AC  D0  0005A         MOVL     SCB, R0                                          ; 3868
     1E          00     6E            00  2C  0005E         MOVC5    #0, (SP), #0, #30, 608(R0)                       ;
                                0260  C0      00063                                                               ;
                                          04  00066         RET                                                       ; 3870

; Routine Size:  103 bytes,    Routine Base:  CODE + 1184
; 2967          3871  1 END
; 2968          3872  0 ELUDOM


                                                                  .EXTRN  LIB$SIGNAL, LIB$STOP

;                                   PSECT SUMMARY
;
;       Name                    Bytes                      Attributes
;
;   CODE                            4587  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                              Library Statistics
;
;                                           -------- Symbols --------         Pages        Processing
;       File                                Total    Loaded   Percent         Mapped       Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1            18619      113         0          1000           00:01.9


;                                   COMMAND QUALIFIERS
;
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:DISPATCH/OBJ=OBJ$:DISPATCH MSRC$:DISPATCH/UPDATE=(ENH$:DISPATCH)

; Size:          4342 code + 245 data bytes
; Run Time:         01:40.9
; Elapsed Time:     04:30.3
; Lines/CPU Min:      2302
; Lexemes/CPU-Min: 24505
; Memory Used:  746 pages
; Compilation Complete
```

0309 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY